```
      program main

      integer,parameter :: maxres=1000
      integer,parameter :: mxratm=200
      integer,parameter :: maxsize=200

      character(len=40) Theory
      integer level
      integer error
      integer cut
      integer numres
      integer ncpu

      Theory='HF/3-21G'
      level=1
      cut=2

      call MFCC(Theory,level,cut,maxres,mxratm,maxsize)

      end
c==================================================================
      subroutine MFCC(Theory,level,cut,maxres,mxratm,maxsize)

c**********************************************************************c
c~~~~~~~~~~~~~~Molecular Fractionation with Conjugate caps~~~~~~~~~~~~~~c
c                                                                      c
c       coord: cartesian coordinates of each atom in each residue      c
c       res_atomnum: the number of atoms in each residue               c
c       res_atomname: the name of the atom in each residue             c
c       res_name: residue name                                         c
c       charge: charge of each residue (only 5 amino acids can have    c
c               a charge and we just double-check them                 c
c       endcharge: check the charge of the terminals (like, NH3 at     c
c                  N-terminal means positive charge and COO at         c
c                  C-terminal means negative charge                    c
c       numres: the total number of residues in the protein            c
c       Theory: method in the ab initio calculation (like, HF, B3LYP...)c
c       BasisSet: basis set used in the ab initio calculation          c
c       cut=0          cut between CA and N                            c
c               level1:         NH2 --- CH3                            c
c               level2:         NH2 --- CH2R                           c
c               level3:       NHCOH --- CH2R                           c
c               level4:         NH2 --- CHRCONH2                       c
c               level5:       NHCOH --- CHRCONH2                       c
c       cut=1          cut between CA and C                            c
c               level1:         CH3 --- CONH2                          c
c               level2:        CH2R --- CONH2                          c
c               level3:      CHRNH2 --- CONH2                          c
c               level4:    CHRNHCOH --- CONH2                          c
c       cut=2          cut CONH (planar Frag)                          c
c               level1:       CH3NH --- COCH3                          c
c               level2:      CH2RNH --- COCH3                          c
c               level3:       CH3NH --- COCH2R                         c
c               level4:      CH2RNH --- COCH2R                         c
c       error: if error=1, we do three-body correction                 c
c              if error=0, no correction                               c
c**********************************************************************c
c**********************************************************************c

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
```

```
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      character(len=40) Theory
      integer level
      integer error
      integer cut
      integer numres
      integer ncpu

      integer maxsize
      integer charge(maxres)
      integer endcharge(2)

      call readpdb(coord,res_atomname,res_atomnum,res_name,
     $               charge,endcharge,numres,maxres,mxratm)

      call calenergy(coord,res_atomname,res_atomnum,res_name,
     $               charge,endcharge,numres,maxres,mxratm,
     $               maxsize,Theory,level,cut)

      call System('rm -f fort.* out* 100* 200*')

      end
c-------------------------------------------------------------
      subroutine calenergy(coord,res_atomname,res_atomnum,
     $               res_name,charge,endcharge,numres,maxres,
     $               mxratm,maxsize,Theory,level,cut)

c**********************************************************************c
c**********************************************************************c
c       atom_symbol: the name of atoms in the small molecule          c
c       ligand: the cartesian coordinates of atoms in molecule        c
c       ligand_atomnum: atom number of ligand molecule                c
c       Frag: the cartesian coordinates of atoms in each fragment     c
c       Frag_atomname: the name of atoms in each fragment             c
c       Frag_atomnum: the number of atoms in each fragment            c
c       Frag_id: the polarity of each residue                         c
c             nopolar(1),polar(-1),polar with charge(0)               c
c       Cap: the cartesian coordinates of atoms in each cap           c
c       Cap_atomname: the name of atoms in each cap                   c
c       Cap_atomnum: the number of atoms in each cap                  c
c       Cap_id: the polarity of each cap                              c
c                                                                     c
c       In correction, we put two nearest residues together           c
c       and treat them as one fragment (like, 12,23,34,45...)         c
c       Corr: the cartesian coordinates of atoms for each fragment    c
c       Corr_atomname: the name of atoms in each corr fragment        c
c       Corr_atomnum: the number of atoms in corr fragment            c
c       pcharge: the charge for each fragment                         c
c       ccharge: the charge for each cap                              c
c       Corr_charge: the charge for each corr fragment                c
c       p_frag: the sum of fragment energies                          c
c       p_cap: the sum of cap energies                                c
c       p_sul: the sum of sulcap energies                             c
c**********************************************************************c
c**********************************************************************c

      integer maxres
      integer mxratm
```

```
        real*8 coord(maxres,mxratm,3)
        character(len=4) res_atomname(maxres,mxratm)
        character(len=4) res_name(maxres)
        integer res_atomnum(maxres)


        integer maxsize

        character(len=4) atom_symbol(maxsize)
        real*8 ligand(maxsize,3)
        integer ligand_atomnum
        integer ligand_charge

        real*8 Frag(maxres,mxratm,3)
        character(len=4) Frag_atomname(maxres,mxratm)
        integer Frag_atomnum(maxres)
        integer Frag_id(maxres)

        real*8 Cap(maxres,mxratm,3)
        character(len=4) Cap_atomname(maxres,mxratm)
        integer Cap_atomnum(maxres)
        integer Cap_id(maxres)

        real*8 Corr(maxres,mxratm,3)
        character(len=4) Corr_atomname(maxres,mxratm)
        integer Corr_atomnum(maxres)
        integer Corr_charge(maxres)
        real*8 SulCap(maxres,mxratm,3)
        character(len=4) SulCap_atomname(maxres,mxratm)
        integer SulCap_atomnum(maxres)
        integer sn
        character(len=40) Theory
        integer level
        integer error
        integer cut
        integer numres
        integer ncpu

        real*8 fgra_lig(maxsize,3)
        real*8 cgra_lig(maxsize,3)
        real*8 sgra_lig(maxsize,3)
        real*8 gra_lig(maxsize,3)

        integer charge(maxres)
        integer endcharge(2)
        integer pcharge(maxres)
        integer ccharge(maxres)
        integer scharge(maxres)

        real*8 f_asymp(maxres)
        real*8 c_asymp(maxres)
        real*8 s_asymp(maxres)
        real*8 fc_poten(maxres)
        real*8 p_frag
        real*8 p_cap
        real*8 p_sul

        integer i
        integer ires

c*----------get the coordinates of small molecule------------*c
        call readligand(atom_symbol,ligand,ligand_atomnum,
     $                   ligand_charge,maxsize)


c*----construct Z matrix of the interaction of full system----*c
c        call exaGauss(coord,res_atomname,numres,res_atomnum,
c     $          maxres,mxratm,-1,atom_symbol,ligand,
c     $          ligand-atomnum,maxsize,500,Theory)


c*-----------cut proteins into N pieces and N-1 caps----------*c
        call cut_protein(coord,res_atomname,res_atomnum,res_name,
     $                   numres,Frag,Frag_atomname,Frag_atomnum,
     $                   Cap,Cap_atomname,Cap_atomnum,maxres,
     $                   mxratm,level,cut)


c*--determine the #s of CYS(CYX) residues & of disulfur bond--*c
        call disulf_bond(coord,res_atomname,res_atomnum,res_name,
     $                   numres,Frag,Frag_atomname,Frag_atomnum,
     $                   SulCap,SulCap_atomname,SulCap_atomnum,
     $                   maxres,mxratm,sn)
c       print*,'*********',sn,' disulfur bond(s) found *********'
c       print*


c*--------determine the charge for each piece and cap---------*c
        call get_pccharge(charge,endcharge,pcharge,ccharge,
     $                   res_name,numres,maxres,level,cut)


c*----------determine the polarity of fragment & cap----------*c
        call polarity(res_name,Frag_id,Cap_id,numres,
     $                   maxres,level,cut)


c*-------calculate ab initio energies for each fragment-------*c
        call getenergy(Frag,Frag_atomname,Frag_atomnum,Frag_id,
     $                   pcharge,maxres,mxratm,atom_symbol,ligand,
     $                   ligand_atomnum,ligand_charge,maxsize,
     $                   numres,Theory,1,f_asymp,p_frag,fgra_lig)


c*---------calculate ab initio energies for each caps---------*c
        call getenergy(Cap,Cap_atomname,Cap_atomnum,Cap_id,
     $                   ccharge,maxres,mxratm,atom_symbol,ligand,
     $                   ligand_atomnum,ligand_charge,maxsize,
     $                   numres,Theory,2,c_asymp,p_cap,cgra_lig)


c*------calculate ab initio energies for disulfur cap(s)------*c
        if(sn.ge.1) then
           do i=1, sn
              scharge(i)=0
              s_asymp(i)=0.d0
           end do
           call getenergy(SulCap,SulCap_atomname,SulCap_atomnum,
     $                   -1,scharge,maxres,mxratm,atom_symbol,
     $                   ligand,ligand_atomnum,ligand_charge,
     $                   maxsize,sn,Theory,1,s_asymp,p_sul,sgra_lig)
        end if


c*--------------calculate total interaction energy------------*c
```

```
      print*, 'M       M  F F F F F    C C C    C C C'
      print*, 'M M   M M  F          C        C      '
      print*, 'M  M M  M  F F F F   C        C       '
      print*, 'M   M   M  F          C        C      '
      print*, 'M       M  F            C C C    C C C'
      print*, 'Normalized interaction energy == ', (p_frag
     $               -p_cap-p_sul)*27.2114d0*23.0605d0, 'kcal/mol'
      print*, 'End of calculation'

      return
      end
c=================================================================
      subroutine getenergy(tmp,tmp_atomname,tmp_atomnum,tmp_id,
     $                charge,maxres,mxratm,atom_symbol,ligand,
     $                ligand_atomnum,ligand_charge,maxsize,
     $                numres,Theory,start,asymp,abc,gradient)

      integer maxres
      integer mxratm

      real*8 tmp(maxres,mxratm,3)
      character(len=4) tmp_atomname(maxres,mxratm)
      integer tmp_atomnum(maxres)
      integer tmp_id(maxres)
      integer maxsize

      character(len=4) atom_symbol(maxsize)
      real*8 ligand(maxsize,3)
      integer ligand_atomnum
      integer ligand_charge

      character(len=40) Theory
      integer level
      integer error
      integer cut
      integer numres
      integer ncpu

      real*8 gradient(maxsize,3)
      integer charge(maxres)
      real*8 asymp(maxres)
      real*8 pes(maxres)
      real*8 abc, p_ligand

      integer id
      integer start
      real*8 min
      real*8 energy
      real*8 asyme

c*---------calculate isolated ligand ab initio energy---------*c
       call get_unit(iout)
       call Gaussian(tmp,tmp_atomname,tmp_atomnum,1,iout,
     $                charge(1),maxres,mxratm,atom_symbol,
     $                ligand,ligand_atomnum,ligand_charge,
     $                maxsize,2,Theory)
       call Abinical(iout,p_ligand,gradient,0,maxsize,number*3,
     $                (number+ligand_atomnum)*3,Theory)
c*---------calculate the normalized interaction energy--------*c
       do ires=start, numres
          call mindis(tmp,tmp_atomnum,tmp_atomname,ires,numres,
     $                ligand,ligand_atomnum,maxsize,maxres,
     $                mxratm,min)
```

```
            call selectgroup(tmp_id,ires,4000.0,4200.0,4600.0,
     $                   min,maxres,id)
            if(id.ne.0) then
               call get_unit(iout)
               call Gaussian(tmp,tmp_atomname,tmp_atomnum,ires,
     $                   iout,charge(ires),maxres,mxratm,atom_symbol,
     $                   ligand,ligand_atomnum,ligand_charge,maxsize,
     $                   1,Theory)
               number=tmp_atomnum(ires)
               call Abinical(iout,energy,gradient,0,maxsize,number*3,
     $                   (number+ligand_atomnum)*3,Theory)
               pes(ires)=energy
c*----calc. the corresponding isolated frag/cap energy once---*c
               if(asymp(ires).eq.0.d0) then
                  call get_unit(iout)
                  call Gaussian(tmp,tmp_atomname,tmp_atomnum,ires,
     $                   iout,charge(ires),maxres,mxratm,atom_symbol,
     $                   ligand,ligand_atomnum,ligand_charge,maxsize,
     $                   0,Theory)
                  call Abinical(iout,energy,gradient,0,maxsize,number*3,
     $                   (number+ligand_atomnum)*3,Theory)
               end if
               pes(ires)=pes(ires)-energy-p_ligand
               abc=abc+pes(ires)
               write(*,*) ires, pes(ires)
            end if
         end do

         return
         end
c===========================================================
         subroutine Abinical(iout,energy,gradient,id,maxsize,start,
     $                             end,Theory)

         integer maxsize
         real*8 energy
         real*8 gradient(maxsize,3)
         integer id
         integer start
         integer end
         integer iout
         integer input_unit
         character(len=50) method
         character(len=100) grep
         character(len=100) cut1
         character(len=100) cut2
         character(len=3) tmp
         character(len=40) Theory
         integer i
         integer k
         integer kk
         integer kkk

         jout=iout

         do j=1, 3
            kkk=jout/10**2
            kk=(jout-kkk*100)/10
            k=(jout-kkk*100-kk*10)
         end do

         tmp=char(48+kkk)//char(48+kk)//char(48+k)
```

```
        method='g98<fort.'//char(48+iout)//'> out'//tmp

        call System(method)

        if(Theory(1:3).ne.'MP2') then
c************************************************************
        grep='grep "SCF Done" out'//tmp//'>100'//tmp
        call System(grep)

        cut1='cut -d"=" -f2 100'//tmp//'>200'//tmp
        call System(cut1)

        cut2='cut -d"A" -f1 200'//tmp//'>100'//tmp
        call System(cut2)

        call get_unit(input_unit)
        open(input_unit,file='100'//tmp,status='old')
        read(input_unit,*) energy
        close(input_unit)
c***********************************************************
        else
        grep='grep "EUMP2" out'//tmp//'>100'//tmp
        call System(grep)

        cut1='cut -d"=" -f3 100'//tmp//'>200'//tmp
        call System(cut1)

        call get_unit(input_unit)
        open(input_unit,file='200'//tmp,status='old')
        read(input_unit,*) energy
        close(input_unit)
c***********************************************************
        end if

        if(id.eq.1) then
c          call findgradient(gradient,start,end,maxsize)
        end if

        return
        end
c==============================================================
        subroutine errcorr(Coord,res_atomname,res_atomnum,charge,
     $                endcharge,maxres,mxratm,atom_symbol,ligand,
     $                ligand_atomnum,ligand_charge,maxsize,
     $                Theory,level,error,fc_poten)

        integer maxres
        integer mxratm

        real*8 coord(maxres,mxratm,3)
        character(len=4) res_atomname(maxres,mxratm)
        character(len=4) res_name(maxres)
        integer res_atomnum(maxres)


        real*8 Corr(maxres,mxratm,3)
        character(len=4) Corr_atomname(maxres,mxratm)
        integer Corr_atomnum(maxres)
        integer Corr_charge(maxres)
        integer maxsize

        character(len=4) atom_symbol(maxsize)
        real*8 ligand(maxsize,3)
```

```
      integer ligand_atomnum
      integer ligand_charge

      character(len=40) Theory
      integer level
      integer error
      integer cut
      integer numres
      integer ncpu

      integer charge(maxres)
      integer endcharge(2)

      integer ires
      integer iout
      real*8 fc_poten(maxres)
      real*8 energy

      do ires=1, numres-1
         call get_unit(iout)
         call combineunits(Coord,res_atomname,res_atomnum,
     $                Corr,Corr_atomname,Corr_atomnum,ires,
     $                level,charge,endcharge,Corr_charge,
     $                maxres,mxratm,numres)
         call Gaussian(Corr,Corr_atomname,Corr_atomnum,ires,
     $                iout,Corr_charge(ires),maxres,mxratm,
     $                atom_symbol,ligand,ligand_atomnum,
     $                ligand_charge,maxsize,1,Theory)
c        call Abinical(energy)
c        fc_poten(ires)=energy
      end do

      return
      end
c=============================================================
      subroutine findgradient(gradient,start,end,maxsize)

      integer maxsize
      integer start
      integer end
      integer input_unit
      character(len=128) string
      character(len=20) var
      real*8 old
      real*8 der
      real*8 gradient(maxsize,3)
      integer n
      logical s_eqi

      call get_unit(input_unit)
      open(unit=input_unit,file='out',status='old')

      do
        read(input_unit,'(a)') string
        if(s_eqi(string(2:9),'Variable')) then
           read(input_unit,'(a)') string
           read(input_unit,'(a)') string
           do i=1, start
              read(input_unit,'(a)') string
           end do
           do i=start+1, end
              n=n+1
              read(string,'(a11,f10.5,f10.5)') var,old,der
```

```
            gradient(1+(n-1)/3,n-3*((n-1)/3))=der
            read(input_unit,'(a)') string
         end do
         return
      end if
    end do

    close(input_unit)

    return
    end
c=================================================================
    subroutine cut_protein(coord,res_atomname,res_atomnum,
   $                res_name,numres,Frag,Frag_atomname,
   $                Frag_atomnum,Cap,Cap_atomname,
   $                Cap_atomnum,maxres,mxratm,level,cut)

    integer maxres
    integer mxratm

    real*8 coord(maxres,mxratm,3)
    character(len=4) res_atomname(maxres,mxratm)
    character(len=4) res_name(maxres)
    integer res_atomnum(maxres)


    real*8 Frag(maxres,mxratm,3)
    character(len=4) Frag_atomname(maxres,mxratm)
    integer Frag_atomnum(maxres)
    integer Frag_id(maxres)

    real*8 Cap(maxres,mxratm,3)
    character(len=4) Cap_atomname(maxres,mxratm)
    integer Cap_atomnum(maxres)
    integer Cap_id(maxres)

    character(len=40) Theory
    integer level
    integer error
    integer cut
    integer numres
    integer ncpu

    integer ires

    do ires=1, numres
c*----------for the first residue in the protein chain--------*c
       if(ires.eq.1) then
          call resno1(coord,res_atomname,res_atomnum,
   $                Frag,Frag_atomname,Frag_atomnum,
   $                res_name,ires,maxres,mxratm,level,cut)


c*----------for the last residues in the protein chain--------*c
       else if(ires.eq.numres) then
            call resend(coord,res_atomname,res_atomnum,
   $                Frag,Frag_atomname,Frag_atomnum,
   $                res_name,ires,maxres,mxratm,level,cut)


c*---------for the middle residues in the protein chain-------*c
       else
          call resmid(coord,res_atomname,res_atomnum,
     $                    Frag,Frag_atomname,Frag_atomnum,res_name,
     $                    ires,numres,maxres,mxratm,level,cut)

            end if


c*-----locate the caps in the middle of the protein chain-----*c
            if(1.lt.ires) then
               call findcap(coord,res_atomname,res_atomnum,
     $                    Cap,Cap_atomname,Cap_atomnum,res_name,
     $                    ires,numres,maxres,mxratm,level,cut)
            end if
         end do

         return
         end
c==================================================================
         subroutine resnol(coord,res_atomname,res_atomnum,
     $                    Frag,Frag_atomname,Frag_atomnum,
     $                    res_name,ires,maxres,mxratm,level,cut)

         integer maxres
         integer mxratm

         real*8 coord(maxres,mxratm,3)
         character(len=4) res_atomname(maxres,mxratm)
         character(len=4) res_name(maxres)
         integer res_atomnum(maxres)


         real*8 Frag(maxres,mxratm,3)
         character(len=4) Frag_atomname(maxres,mxratm)
         integer Frag_atomnum(maxres)
         integer Frag_id(maxres)

         character(len=40) Theory
         integer level
         integer error
         integer cut
         integer numres
         integer ncpu
         integer i
         integer j
         integer iatom
         integer k
         integer step
         integer ires
         integer jres
         real*8 x(3)
         real*8 y(3)
         real*8 y1(3)
         character(len=4) A1
         character(len=4) A2
         character(len=4) AA


c*--------copy the current residue: center of fragment--------*c
         call current_residue(coord,res_atomname,res_atomnum,
     $                    Frag,Frag_atomname,Frag_atomnum,
     $                    ires,maxres,mxratm)
```

```
c*-----------------construct the end cap(s)-------------------*c
        step=0
cssssssssssssssssss cut=0 i.e. cut CA-N bond ssssssssssssssssssc
        if(cut.eq.0) then
           if(res_name(ires+1).ne.'PRO') then
              if(level.eq.1) then
                 call CH3next(coord,res_atomname,res_atomnum,ires,
     $                  Frag,Frag_atomname,Frag_atomnum,ires,
     $                  maxres,mxratm,step)
              end if

              if(level.eq.2.or.level.eq.3) then
                 call CH2Rnext(coord,res_atomname,res_atomnum,ires,
     $                  Frag,Frag_atomname,Frag_atomnum,ires,
     $                  maxres,mxratm,step)
              end if

              if(level.eq.4.or.level.eq.5) then
                 call CONHnext(coord,res_atomname,res_atomnum,ires,
     $                  Frag,Frag_atomname,Frag_atomnum,ires,
     $                  maxres,mxratm,step)
              end if
           else
              call CH2Rnext(coord,res_atomname,res_atomnum,ires,
     $                  Frag,Frag_atomname,Frag_atomnum,ires,
     $                  maxres,mxratm,step)
           end if
        end if
cssssssssssssssssss cut=1 i.e. cut CA-C bond ssssssssssssssssssc
        if(cut.eq.1) then
           if(res_name(ires+1).ne.'PRO') then
              call NH2next(coord,res_atomname,res_atomnum,ires,
     $                  Frag,Frag_atomname,Frag_atomnum,ires,
     $                  maxres,mxratm,step)
           else
              call CH2Rnext(coord,res_atomname,res_atomnum,ires,
     $                  Frag,Frag_atomname,Frag_atomnum,ires,
     $                  maxres,mxratm,step)
           end if
        end if
cssssssssssssssssss cut=2 i.e. cut C-N bond sssssssssssssssssssc
        if(cut.eq.2) then
           if(res_name(ires+1).ne.'PRO') then
              if(level.eq.1.or.level.eq.3) then
                 call CH3next(coord,res_atomname,res_atomnum,ires,
     $                  Frag,Frag_atomname,Frag_atomnum,ires,
     $                  maxres,mxratm,step)
              else
                 call CH2Rnext(coord,res_atomname,res_atomnum,ires,
     $                  Frag,Frag_atomname,Frag_atomnum,ires,
     $                  maxres,mxratm,step)
              end if
           else
              call CH2Rnext(coord,res_atomname,res_atomnum,ires,
     $                  Frag,Frag_atomname,Frag_atomnum,ires,
     $                  maxres,mxratm,step)
           end if
        end if
csssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss

        return
        end
c===============================================================
      subroutine resend(coord,res_atomname,res_atomnum,
     $                  Frag,Frag_atomname,Frag_atomnum,
     $                  res_name,ires,maxres,mxratm,level,cut)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Frag(maxres,mxratm,3)
      character(len=4) Frag_atomname(maxres,mxratm)
      integer Frag_atomnum(maxres)
      integer Frag_id(maxres)

      character(len=40) Theory
      integer level
      integer error
      integer cut
      integer numres
      integer ncpu
      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


c*--------copy the current residue: center of fragment--------*c
      call current_residue(coord,res_atomname,res_atomnum,
     $                Frag,Frag_atomname,Frag_atomnum,
     $                ires,maxres,mxratm)


c*-----------------construct the end cap(s)-------------------*c
      step=0
cssssssssssssssssss cut=0 i.e. cut CA-N bond ssssssssssssssssssc
      if(cut.eq.0) then
         if(level.eq.1.or.level.eq.2.or.level.eq.4) then
            call NH2prev(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,ires,
     $                maxres,mxratm,step)
         end if

         if(level.eq.3.or.level.eq.5) then
            call NHCOprev(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,ires,
     $                maxres,mxratm,step)
         end if
      end if
cssssssssssssssssss cut=1 i.e. cut CA-C bond ssssssssssssssssssc
```

```
      if(cut.eq.1) then
         if(res_name(ires-1).ne.'PRO') then
            if(level.eq.1) then
               call CH3prev(coord,res_atomname,res_atomnum,ires,
     $                 Frag,Frag_atomname,Frag_atomnum,ires,
     $                 maxres,mxratm,step)
            end if

            if(level.eq.2) then
               call CH2Rprev(coord,res_atomname,res_atomnum,ires,
     $                 Frag,Frag_atomname,Frag_atomnum,ires,
     $                 maxres,mxratm,step)
            end if

            if(level.eq.3) then
               call CHRNH2prev(coord,res_atomname,res_atomnum,ires,
     $                 Frag,Frag_atomname,Frag_atomnum,ires,
     $                 maxres,mxratm,step)
            end if

            if(level.eq.4.or.level.eq.5) then
               call CHRNHCOHprev(coord,res_atomname,res_atomnum,ires,
     $                 Frag,Frag_atomname,Frag_atomnum,ires,
     $                 maxres,mxratm,step)
            end if
         else
            call CHRNHCOHprev(coord,res_atomname,res_atomnum,ires,
     $                 Frag,Frag_atomname,Frag_atomnum,ires,
     $                 maxres,mxratm,step)
         end if
      end if
csssssssssssssssss cut=2 i.e. cut C-N bond ssssssssssssssssssc
      if(cut.eq.2) then
         if(res_name(ires-1).ne.'PRO') then
            if(level.eq.1.or.level.eq.2) then
               call CH3prev(coord,res_atomname,res_atomnum,ires,
     $                 Frag,Frag_atomname,Frag_atomnum,ires,
     $                 maxres,mxratm,step)
            else
               call CH2Rprev(coord,res_atomname,res_atomnum,ires,
     $                 Frag,Frag_atomname,Frag_atomnum,ires,
     $                 maxres,mxratm,step)
            end if
         else
            call CHRNHCOHprev(coord,res_atomname,res_atomnum,ires,
     $                 Frag,Frag_atomname,Frag_atomnum,ires,
     $                 maxres,mxratm,step)
         end if
      end if
cssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss

      return
      end
c=============================================================
      subroutine resmid(coord,res_atomname,res_atomnum,
     $         Frag,Frag_atomname,Frag_atomnum,res_name,
     $         ires,numres,maxres,mxratm,level,cut)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
```

```
        character(len=4) res_name(maxres)
        integer res_atomnum(maxres)


        real*8 Frag(maxres,mxratm,3)
        character(len=4) Frag_atomname(maxres,mxratm)
        integer Frag_atomnum(maxres)
        integer Frag_id(maxres)

        character(len=40) Theory
        integer level
        integer error
        integer cut
        integer numres
        integer ncpu
        integer i
        integer j
        integer iatom
        integer k
        integer step
        integer ires
        integer jres
        real*8 x(3)
        real*8 y(3)
        real*8 y1(3)
        character(len=4) A1
        character(len=4) A2
        character(len=4) AA



c*--------copy the current residue: center of fragment--------*c
        call current_residue(coord,res_atomname,res_atomnum,
     $                 Frag,Frag_atomname,Frag_atomnum,
     $                 ires,maxres,mxratm)


c*-----------------construct the end cap(s)-------------------*c
        step=0
csssssssssssssssss cut=0 i.e. cut CA-N bond sssssssssssssssssc
        if(cut.eq.0) then
           if(level.eq.1) then
              call NH2prev(coord,res_atomname,res_atomnum,ires,
     $                 Frag,Frag_atomname,Frag_atomnum,ires,
     $                 maxres,mxratm,step)
              if(res_name(ires+1).ne.'PRO') then
                 call CH3next(coord,res_atomname,res_atomnum,ires,
     $                 Frag,Frag_atomname,Frag_atomnum,ires,
     $                 maxres,mxratm,step)
              else
                 call CH2Rnext(coord,res_atomname,res_atomnum,ires,
     $                 Frag,Frag_atomname,Frag_atomnum,ires,
     $                 maxres,mxratm,step)
              end if
           end if

           if(level.eq.2) then
              call NH2prev(coord,res_atomname,res_atomnum,ires,
     $                 Frag,Frag_atomname,Frag_atomnum,ires,
     $                 maxres,mxratm,step)
              call CH2Rnext(coord,res_atomname,res_atomnum,ires,
     $                 Frag,Frag_atomname,Frag_atomnum,ires,
     $                 maxres,mxratm,step)
```

```
         end if

         if(level.eq.3) then
            call NHCOprev(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,ires,
     $                maxres,mxratm,step)
            call CH2Rnext(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,ires,
     $                maxres,mxratm,step)
         end if

         if(level.eq.4) then
            call NH2prev(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,ires,
     $                maxres,mxratm,step)
            if(ires.eq.(numres-1)) then
               call CH2Rnext(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,ires,
     $                maxres,mxratm,step)
            else
               call CONHnext(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,ires,
     $                maxres,mxratm,step)
            end if
         end if

         if(level.eq.5) then
            call NHCOprev(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,ires,
     $                maxres,mxratm,step)
            if(ires.eq.(numres-1)) then
               call CH2Rnext(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,ires,
     $                maxres,mxratm,step)
            else
               call CONHnext(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,ires,
     $                maxres,mxratm,step)
            end if
         end if
      end if
cssssssssssssssssss cut=1 i.e. cut CA-C bond ssssssssssssssssssc
      if(cut.eq.1) then
         if(res_name(ires+1).ne.'PRO') then
            call NH2next(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,ires,
     $                maxres,mxratm,step)
         else
            call CH2Rnext(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,ires,
     $                maxres,mxratm,step)
         end if

         if(res_name(ires-1).ne.'PRO') then
            if(level.eq.1) then
               call CH3prev(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,ires,
     $                maxres,mxratm,step)
            end if

            if(level.eq.2) then
               call CH2Rprev(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,ires,
     $             maxres,mxratm,step)
          end if

          if(level.eq.3) then
            if(ires.eq.2) then
              call CH2Rprev(coord,res_atomname,res_atomnum,ires,
     $              Frag,Frag_atomname,Frag_atomnum,ires,
     $              maxres,mxratm,step)
            else
              call CHRNH2prev(coord,res_atomname,res_atomnum,ires,
     $              Frag,Frag_atomname,Frag_atomnum,ires,
     $              maxres,mxratm,step)
            end if
          end if

          if(level.eq.4.or.level.eq.5) then
            if(ires.eq.2) then
              call CH2Rprev(coord,res_atomname,res_atomnum,ires,
     $              Frag,Frag_atomname,Frag_atomnum,ires,
     $              maxres,mxratm,step)
            else
              call CHRNHCOHprev(coord,res_atomname,res_atomnum,ires,
     $              Frag,Frag_atomname,Frag_atomnum,ires,
     $              maxres,mxratm,step)
            end if
          end if
       else if(ires.eq.2) then
          call CH3prev(coord,res_atomname,res_atomnum,ires,
     $              Frag,Frag_atomname,Frag_atomnum,ires,
     $              maxres,mxratm,step)
       else if(ires.gt.2) then
          call CHRNHCOHprev(coord,res_atomname,res_atomnum,ires,
     $              Frag,Frag_atomname,Frag_atomnum,ires,
     $              maxres,mxratm,step)
       end if
    end if
csssssssssssssssss cut=2 i.e. cut C-N bond ssssssssssssssssssc
    if(cut.eq.2) then
       if(res_name(ires+1).ne.'PRO') then
          if(level.eq.1.or.level.eq.3) then
             call CH3next(coord,res_atomname,res_atomnum,ires,
     $              Frag,Frag_atomname,Frag_atomnum,ires,
     $              maxres,mxratm,step)
          else
             call CH2Rnext(coord,res_atomname,res_atomnum,ires,
     $              Frag,Frag_atomname,Frag_atomnum,ires,
     $              maxres,mxratm,step)
          end if
       else
          call CH2Rnext(coord,res_atomname,res_atomnum,ires,
     $              Frag,Frag_atomname,Frag_atomnum,ires,
     $              maxres,mxratm,step)
       end if

       if(res_name(ires-1).ne.'PRO') then
          if(level.eq.1.or.level.eq.2) then
             call CH3prev(coord,res_atomname,res_atomnum,ires,
     $              Frag,Frag_atomname,Frag_atomnum,ires,
     $              maxres,mxratm,step)
          else
             call CH2Rprev(coord,res_atomname,res_atomnum,ires,
     $              Frag,Frag_atomname,Frag_atomnum,ires,
     $              maxres,mxratm,step)
```

```
            end if
         else if(ires.eq.2) then
            call CH3prev(coord,res_atomname,res_atomnum,ires,
     $               Frag,Frag_atomname,Frag_atomnum,ires,
     $               maxres,mxratm,step)
         else if(ires.gt.2) then
            call CHRNHCOHprev(coord,res_atomname,res_atomnum,ires,
     $               Frag,Frag_atomname,Frag_atomnum,ires,
     $               maxres,mxratm,step)
         end if
      end if
csssssssssssssssssssssssssssssssssssssssssssssssssssssssssss

      return
      end
c===========================================================
      subroutine findcap(coord,res_atomname,res_atomnum,
     $               Cap,Cap_atomname,Cap_atomnum,res_name,
     $               ires,numres,maxres,mxratm,level,cut)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Cap(maxres,mxratm,3)
      character(len=4) Cap_atomname(maxres,mxratm)
      integer Cap_atomnum(maxres)
      integer Cap_id(maxres)

      character(len=40) Theory
      integer level
      integer error
      integer cut
      integer numres
      integer ncpu
      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      step=0
cssssssssssssssssss cut=0 i.e. cut CA-N bond sssssssssssssssssc
      if(cut.eq.0) then
         if(level.eq.1) then
            if(res_name(ires).ne.'PRO') then
               call Cap_CH3(coord,res_atomname,res_atomnum,
     $               ires,Cap,Cap_atomname,Cap_atomnum,
     $               maxres,mxratm,step)
```

```
            else
               call Cap_CH2R(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
            end if

            call Cap_NH2(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
         end if

         if(level.eq.2) then
            call Cap_CH2R(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
            call Cap_NH2(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
         end if

         if(level.eq.3) then
            call Cap_CH2R(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
            call Cap_NHCO(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
         end if

         if(level.eq.4) then
            if(ires.eq.numres) then
               call Cap_CH2R(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
            else
               call Cap_CONH(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
            end if
            call Cap_NH2(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
         end if

         if(level.eq.5) then
            if(ires.eq.numres) then
               call Cap_CH2R(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
            else
               call Cap_CONH(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
            end if
            call Cap_NHCO(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
         end if
      end if
csssssssssssssssss cut=1 i.e. cut CA-C bond sssssssssssssssssc
      if(cut.eq.1) then
         if(res_name(ires).ne.'PRO') then
            call ProCap_NH2(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                maxres,mxratm,step)
           else
              call Cap_CH2R(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                maxres,mxratm,step)
           end if

           if(res_name(ires-1).ne.'PRO') then
              if(level.eq.1) then
                 call ProCap_CH3(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                maxres,mxratm,step)
              end if

              if(level.eq.2) then
                 call ProCap_CH2R(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                maxres,mxratm,step)
              end if

              if(level.eq.3) then
                if(ires.eq.2) then
                  call ProCap_CH2R(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                maxres,mxratm,step)
                else
                  call ProCap_CHRNH2(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                maxres,mxratm,step)
                end if
              end if

              if(level.eq.4.or.level.eq.5) then
                if(ires.eq.2) then
                  call ProCap_CH2R(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                maxres,mxratm,step)
                else
                  call ProCap_CHRNHCOH(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                maxres,mxratm,step)
                end if
              end if
           else if(ires.eq.2) then
              call ProCap_CH3(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                CapC,CapC_atomname,CapC_atomnum,
     $                maxres,mxratm,step)
           else if(ires.gt.2) then
              call ProCap_CHRNHCOH(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                maxres,mxratm,step)
           end if
        end if
csssssssssssssssss cut=2 i.e. cut C-N bond sssssssssssssssssssc
        if(cut.eq.2) then
           if(res_name(ires).ne.'PRO') then
              if(level.eq.1.or.level.eq.3) then
                 call Cap_CH3(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                maxres,mxratm,step)
              else
```

```
               call Cap_CH2R(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
             end if
          else
             call Cap_CH2R(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
          end if

          if(res_name(ires-1).ne.'PRO') then
             if(level.eq.1.or.level.eq.2) then
                call ProCap_CH3(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
             else
                call ProCap_CH2R(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
             end if
          else if(ires.eq.2) then
             call ProCap_CH3(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
          else if(ires.gt.2) then
             call ProCap_CHRNHCOH(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)
          end if
       end if
csssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
       Cap_atomnum(ires)=step

       return
       end
c===============================================================
       subroutine disulf_bond(coord,res_atomname,res_atomnum,
     $          res_name,numres,Frag,Frag_atomname,Frag_atomnum,
     $          SulCap,SulCap_atomname,SulCap_atomnum,
     $          maxres,mxratm,sn)

       integer maxres
       integer mxratm

       real*8 coord(maxres,mxratm,3)
       character(len=4) res_atomname(maxres,mxratm)
       character(len=4) res_name(maxres)
       integer res_atomnum(maxres)


       real*8 Frag(maxres,mxratm,3)
       character(len=4) Frag_atomname(maxres,mxratm)
       integer Frag_atomnum(maxres)
       integer Frag_id(maxres)

       real*8 SulCap(maxres,mxratm,3)
       character(len=4) SulCap_atomname(maxres,mxratm)
       integer SulCap_atomnum(maxres)
       integer sn

       integer numres
       integer num
       real*8 x1(3)
```

```
      real*8 x2(3)
      real*8 dis
      real*8 sulfur(maxres,3)
      integer flag(maxres)

      sn=0
      num=0
      do ires=1, numres
         if(res_name(ires).eq.'CYX'.or.res_name(ires).eq.'CYS') then
            do iatom=1, res_atomnum(ires)
               if(res_atomname(ires,iatom).eq.'SG') then
                  num=num+1
                  flag(num)=ires
                  do k=1, 3
                     sulfur(num,k)=coord(ires,iatom,k)
                  end do
               end if
            end do
         end if
      end do

      if(num.ge.2) then
         do i=1, num
            do j=1, num
               if(i.ne.j.and.i.lt.j) then
                  do k=1, 3
                     x1(k)=sulfur(i,k)
                     x2(k)=sulfur(j,k)
                  end do
                  dis=0.d0
                  call distance(x1,x2,dis)
                  if(dis.lt.2.5d0) then
                     call coordupdate(coord,res_atomname,res_atomnum,
     $                         Frag,Frag_atomname,Frag_atomnum,
     $                         SulCap,SulCap_atomname,SulCap_atomnum,
     $                         flag(i),flag(j),maxres,mxratm,sn)

                  end if
               end if
            end do
         end do
      end if

      return
      end
C==============================================================
      subroutine coordupdate(coord,res_atomname,res_atomnum,
     $                  Frag,Frag_atomname,Frag_atomnum,
     $                  SulCap,SulCap_atomname,SulCap_atomnum,
     $                  m,n,maxres,mxratm,sn)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Frag(maxres,mxratm,3)
      character(len=4) Frag_atomname(maxres,mxratm)
      integer Frag_atomnum(maxres)
      integer Frag_id(maxres)

      real*8 SulCap(maxres,mxratm,3)
      character(len=4) SulCap_atomname(maxres,mxratm)
      integer SulCap_atomnum(maxres)
      integer sn
      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      real*8 tmp1(maxres,mxratm,3)
      character(len=4) tmp1_atomname(maxres,mxratm)
      real*8 tmp2(maxres,mxratm,3)
      character(len=4) tmp2_atomname(maxres,mxratm)

      integer m
      integer n
      integer flag(maxres)

      do i=1, 2
         if(i.eq.1) then
            ires=m
            jres=n
         else
            ires=n
            jres=m
         end if
         A1='CB'
         A2='CA'
         AA='CC'
         call twopoints(coord,res_atomnum,res_atomname,
     $                jres,jres,A1,A2,maxres,mxratm,x,y)
         call bondlength(x,y,AA)
         step=0
         do iatom=1, res_atomnum(jres)
            if(res_atomname(jres,iatom).eq.'SG'.or.
     $         res_atomname(jres,iatom).eq.'CB'.or.
     $         res_atomname(jres,iatom).eq.'HB'.or.
     $         res_atomname(jres,iatom).eq.'CA') then
              step=step+1
              do k=1, 3
                 if(i.eq.1) tmp1(ires,step,k)=coord(jres,iatom,k)
                 if(i.eq.2) tmp2(ires,step,k)=coord(jres,iatom,k)
              end do
              if(res_atomname(jres,iatom).eq.'CA') then
                 if(i.eq.1) tmp1_atomname(ires,step)='H'
                 if(i.eq.2) tmp2_atomname(ires,step)='H'
                 do k=1, 3
                    if(i.eq.1) tmp1(ires,step,k)=y(k)
                    if(i.eq.2) tmp2(ires,step,k)=y(k)
                 end do
              else
```

```
                  if(i.eq.1) tmp1_atomname(ires,step)=
     $                         res_atomname(jres,iatom)
                  if(i.eq.2) tmp2_atomname(ires,step)=
     $                         res_atomname(jres,iatom)
               end if
            end if
         end do
      end do

      do i=1, step
         do k=1, 3
            Frag(m,Frag_atomnum(m)+i,k)=tmp1(m,i,k)
            Frag(n,Frag_atomnum(n)+i,k)=tmp2(n,i,k)
         end do
         Frag_atomname(m,Frag_atomnum(m)+i)=tmp1_atomname(m,i)
         Frag_atomname(n,Frag_atomnum(n)+i)=tmp2_atomname(n,i)
      end do
      Frag_atomnum(m)=Frag_atomnum(m)+step
      Frag_atomnum(n)=Frag_atomnum(n)+step

      sn=sn+1
      do i=1, step
         do k=1, 3
            SulCap(sn,i,k)=tmp1(m,i,k)
         end do
         SulCap_atomname(sn,i)=tmp1_atomname(m,i)
      end do
      do i=1, step
         do k=1, 3
            SulCap(sn,step+i,k)=tmp2(n,i,k)
         end do
         SulCap_atomname(sn,step+i)=tmp2_atomname(n,i)
      end do
      SulCap_atomnum(sn)=step+step

      return
      end
c=================================================================
c=================================================================
      subroutine readligand(atom_symbol,ligand,ligand_atomnum,
     $                 ligand_charge,maxsize)

      integer maxsize

      character(len=4) atom_symbol(maxsize)
      real*8 ligand(maxsize,3)
      integer ligand_atomnum
      integer ligand_charge


      open(111,file='ligand.dat',status='old')
      read(111,*) ligand_atomnum, ligand_charge
      do i=1, ligand_atomnum
         read(111,*) atom_symbol(i),(ligand(i,j),j=1,3)
      end do
      close(111)

      return
      end
c=================================================================
      subroutine exaGauss(coord,res_atomname,numres,res_atomnum,
     $        maxres,mxratm,charge,atom_symbol,ligand,
     $        ligand_atomnum,maxsize,iout,Theory)
```

```
      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      integer maxsize

      character(len=4) atom_symbol(maxsize)
      real*8 ligand(maxsize,3)
      integer ligand_atomnum
      integer ligand_charge

      character(len=40) Theory
      integer level
      integer error
      integer cut
      integer numres
      integer ncpu

      integer charge
      integer ires
      integer iatom
      integer iout

      write(iout,"('#T',2x,a20,2x,'SCF(MaxCycle=500,
     $        Conver=5) test')")      Theory
      write(iout,*)
      write(iout,"('ab initio calculation for full system')")
      write(iout,*)
      write(iout,"(I2,1x,'1')") charge

      do ires=1, numres
         do iatom=1, res_atomnum(ires)
            write(iout,111) res_atomname(ires,iatom)(1:1),
     $                      coord(ires,iatom,1:3)
111         format(a,2x,3g14.6)
         end do
      end do

      do i=1, ligand_atomnum
         write(iout,111) atom_symbol(i), ligand(i,1:3)
      end do

      write(iout,*)
      close(iout)

      return
      end
C=============================================================
      subroutine Gaussian(tmp,tmp_atomname,tmp_atomnum,ires,iout,
     $                charge,maxres,mxratm,atom_symbol,ligand,
     $                ligand_atomnum,ligand_charge,maxsize,id,
     $                Theory)

      integer maxres
      integer mxratm

      real*8 tmp(maxres,mxratm,3)
      character(len=4) tmp_atomname(maxres,mxratm)
      integer tmp_atomnum(maxres)
      integer tmp_id(maxres)
      integer maxsize

      character(len=4) atom_symbol(maxsize)
      real*8 ligand(maxsize,3)
      integer ligand_atomnum
      integer ligand_charge

      character(len=40) Theory
      integer level
      integer error
      integer cut
      integer numres
      integer ncpu

      integer tmp_charge
      integer id
      integer charge
      integer iout
      integer j

      tmp_charge=ligand_charge
      numatm=tmp_atomnum(ires)

      write(iout,"('#T',2x,a21,2x,
     $        'NoSymm SCF(MaxCycle=500,Conver=5) test')") Theory
      write(iout,*)
      write(iout,"('ligand+peptide potential')")
      write(iout,*)

      if(id.ne.2) then
         if(id.eq.0) tmp_charge=0
         write(iout,"(I2,1x,'1')") charge+tmp_charge

         do j=1, numatm
            write(iout,111) tmp_atomname(ires,j),tmp(ires,j,1:3)
111         format(a4,2x,3g14.6)
         end do

         if(id.eq.1) then
            do j=1, ligand_atomnum
               write(iout,111) atom_symbol(j),ligand(j,1:3)
            end do
         end if
      else
         write(iout,"(I2,1x,'1')") tmp_charge
         do j=1, ligand_atomnum
            write(iout,111) atom_symbol(j),ligand(j,1:3)
         end do
      end if

      write(iout,*)
c     write(iout,"('--link1--')")
      close(iout)

      return
      end
c-------------------------------------------------------------
c=============================================================
      subroutine NH2next(coord,res_atomname,res_atomnum,ires,
     $               Frag,Frag_atomname,Frag_atomnum,jres,
$                    maxres,mxratm,step)

   integer maxres
   integer mxratm

   real*8 coord(maxres,mxratm,3)
   character(len=4) res_atomname(maxres,mxratm)
   character(len=4) res_name(maxres)
   integer res_atomnum(maxres)


   real*8 Frag(maxres,mxratm,3)
   character(len=4) Frag_atomname(maxres,mxratm)
   integer Frag_atomnum(maxres)
   integer Frag_id(maxres)

   integer i
   integer j
   integer iatom
   integer k
   integer step
   integer ires
   integer jres
   real*8 x(3)
   real*8 y(3)
   real*8 y1(3)
   character(len=4) A1
   character(len=4) A2
   character(len=4) AA


   A1='N'
   A2='CA'
   AA='CN'
   call twopoints(coord,res_atomnum,res_atomname,
$          ires+1,ires+1,A1,A2,maxres,mxratm,x,y)
   call bondlength(x,y,AA)
   step=0
   do iatom=1, res_atomnum(ires+1)
      if(res_atomname(ires+1,iatom).eq.'N'.or.
$          res_atomname(ires+1,iatom).eq.'H'.or.
$          res_atomname(ires+1,iatom).eq.'CA') then
           step=step+1
        do k=1, 3
            Frag(jres,Frag_atomnum(jres)+step,k)=
$                  coord(ires+1,iatom,k)
        end do
        if(res_atomname(ires+1,iatom).eq.'CA') then
          Frag_atomname(jres,Frag_atomnum(jres)+step)='H'
          do k=1, 3
             Frag(jres,Frag_atomnum(jres)+step,k)=y(k)
          end do
        else
          Frag_atomname(jres,Frag_atomnum(jres)+step)=
$                  res_atomname(ires+1,iatom)(1:1)
        end if
      end if
   end do

   Frag_atomnum(jres)=Frag_atomnum(jres)+step

   return
   end
```

```
C==============================================================
      subroutine CH3prev(coord,res_atomname,res_atomnum,ires,
     $                 Frag,Frag_atomname,Frag_atomnum,jres,
     $                 maxres,mxratm,step)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Frag(maxres,mxratm,3)
      character(len=4) Frag_atomname(maxres,mxratm)
      integer Frag_atomnum(maxres)
      integer Frag_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='CA'
      A2='N'
      AA='CC'
      call twopoints(coord,res_atomnum,res_atomname,
     $        ires-1,ires-1,A1,A2,maxres,mxratm,x,y)
      call bondlength(x,y,AA)
      A1='CA'
      A2='CB'
      call twopoints(coord,res_atomnum,res_atomname,
     $        ires-1,ires-1,A1,A2,maxres,mxratm,x,y1)
      call bondlength(x,y1,AA)
      step=0
      do iatom=1, res_atomnum(ires-1)
         if(res_atomname(ires-1,iatom).eq.'C'.or.
     $        res_atomname(ires-1,iatom).eq.'O'.or.
     $        res_atomname(ires-1,iatom).eq.'CA'.or.
     $        res_atomname(ires-1,iatom).eq.'HA'.or.
     $        res_atomname(ires-1,iatom).eq.'HA1'.or.
     $        res_atomname(ires-1,iatom).eq.'CB'.or.
     $        res_atomname(ires-1,iatom).eq.'HA2'.or.
     $        res_atomname(ires-1,iatom).eq.'HA3'.or.
     $        res_atomname(ires-1,iatom).eq.'N') then
            step=step+1
            do k=1, 3
               Frag(jres,Frag_atomnum(jres)+step,k)=
     $               coord(ires-1,iatom,k)
            end do
            if(res_atomname(ires-1,iatom).eq.'N') then
              Frag_atomname(jres,Frag_atomnum(jres)+step)='H'
            do k=1, 3
               Frag(jres,Frag_atomnum(jres)+step,k)=y(k)
            end do
          end if
          if(res_atomname(ires-1,iatom).eq.'CB') then
            Frag_atomname(jres,Frag_atomnum(jres)+step)='H'
            do k=1, 3
               Frag(jres,Frag_atomnum(jres)+step,k)=y1(k)
            end do
          end if
          if(res_atomname(ires-1,iatom).ne.'N'.and.
     $       res_atomname(ires-1,iatom).ne.'CB') then
            Frag_atomname(jres,Frag_atomnum(jres)+step)=
     $              res_atomname(ires-1,iatom)(1:1)
          end if
        end if
      end do

      Frag_atomnum(jres)=Frag_atomnum(jres)+step

      return
      end
c=================================================================
      subroutine CH2Rprev(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,jres,
     $                maxres,mxratm,step)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Frag(maxres,mxratm,3)
      character(len=4) Frag_atomname(maxres,mxratm)
      integer Frag_atomnum(maxres)
      integer Frag_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='CA'
      A2='N'
      AA='CC'
      call twopoints(coord,res_atomnum,res_atomname,
     $        ires-1,ires-1,A1,A2,maxres,mxratm,x,y)
      call bondlength(x,y,AA)
      step=0
      do iatom=1, res_atomnum(ires-1)
         if(res_atomname(ires-1,iatom).ne.'H'.and.
     $       res_atomname(ires-1,iatom).ne.'H1'.and.
     $       res_atomname(ires-1,iatom).ne.'H2'.and.
     $       res_atomname(ires-1,iatom).ne.'H3') then
           step=step+1
           do k=1, 3
              Frag(jres,Frag_atomnum(jres)+step,k)=
     $               coord(ires-1,iatom,k)
           end do
           if(res_atomname(ires-1,iatom).eq.'N') then
             Frag_atomname(jres,Frag_atomnum(jres)+step)='H'
             do k=1, 3
                Frag(jres,Frag_atomnum(jres)+step,k)=y(k)
             end do
           else
             Frag_atomname(jres,Frag_atomnum(jres)+step)=
     $               res_atomname(ires-1,iatom)(1:1)
           end if
         end if
      end do

      Frag_atomnum(jres)=Frag_atomnum(jres)+step

      return
      end
C=============================================================
      subroutine CHRNH2prev(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,jres,
     $                maxres,mxratm,step)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Frag(maxres,mxratm,3)
      character(len=4) Frag_atomname(maxres,mxratm)
      integer Frag_atomnum(maxres)
      integer Frag_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='N'
      A2='C'
      AA='CN'
      call twopoints(coord,res_atomnum,res_atomname,
     $         ires-1,ires-2,A1,A2,maxres,mxratm,x,y)
      call bondlength(x,y,AA)
      step=0
      do iatom=1, res_atomnum(ires-1)
         step=step+1
         do k=1, 3
            Frag(jres,Frag_atomnum(jres)+step,k)=
     $                  coord(ires-1,iatom,k)
         end do
         Frag_atomname(jres,Frag_atomnum(jres)+step)=
     $                  res_atomname(ires-1,iatom)(1:1)
      end do

      do iatom=1, res_atomnum(ires-2)
         if(res_atomname(ires-2,iatom).eq.'C') then
            step=step+1
            do k=1, 3
               Frag(jres,Frag_atomnum(jres)+step,k)=y(k)
            end do
            Frag_atomname(jres,Frag_atomnum(jres)+step)='H'
         end if
      end do

      Frag_atomnum(jres)=Frag_atomnum(jres)+step

      return
      end
c==========================================================
      subroutine CHRNHCOHprev(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,jres,
     $                maxres,mxratm,step)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Frag(maxres,mxratm,3)
      character(len=4) Frag_atomname(maxres,mxratm)
      integer Frag_atomnum(maxres)
      integer Frag_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='C'
      A2='CA'
      AA='CC'
      call twopoints(coord,res_atomnum,res_atomname,
     $         ires-2,ires-2,A1,A2,maxres,mxratm,x,y)
      call bondlength(x,y,AA)
      step=0
      do iatom=1, res_atomnum(ires-1)
         step=step+1
         do k=1, 3
            Frag(jres,Frag_atomnum(jres)+step,k)=
     $                 coord(ires-1,iatom,k)
         end do
         Frag_atomname(jres,Frag_atomnum(jres)+step)=
     $                 res_atomname(ires-1,iatom)(1:1)
      end do

      do iatom=1, res_atomnum(ires-2)
         if(res_atomname(ires-2,iatom).eq.'C'.or.
     $         res_atomname(ires-2,iatom).eq.'O'.or.
     $         res_atomname(ires-2,iatom).eq.'CA') then
            step=step+1
            do k=1, 3
               Frag(jres,Frag_atomnum(jres)+step,k)=
     $                 coord(ires-2,iatom,k)
            end do
            if(res_atomname(ires-2,iatom).eq.'CA') then
              Frag_atomname(jres,Frag_atomnum(jres)+step)='H'
              do k=1, 3
                 Frag(jres,Frag_atomnum(jres)+step,k)=y(k)
              end do
            else
               Frag_atomname(jres,Frag_atomnum(jres)+step)=
     $                 res_atomname(ires-2,iatom)(1:1)
            end if
         end if
      end do

      Frag_atomnum(jres)=Frag_atomnum(jres)+step

      return
      end
C================================================================
      subroutine ProCap_NH2(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Cap(maxres,mxratm,3)
      character(len=4) Cap_atomname(maxres,mxratm)
      integer Cap_atomnum(maxres)
      integer Cap_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='N'
      A2='CA'
      AA='CN'
      call twopoints(coord,res_atomnum,res_atomname,
     $        ires,ires,A1,A2,maxres,mxratm,x,y)
      call bondlength(x,y,AA)

      do iatom=1, res_atomnum(ires)
         if(res_atomname(ires,iatom).eq.'N'.or.
     $        res_atomname(ires,iatom).eq.'H'.or.
     $        res_atomname(ires,iatom).eq.'CA') then
c     $        res_atomname(ires,iatom).eq.'CB') then
            step=step+1
            do k=1, 3
               Cap(ires,step,k)=coord(ires,iatom,k)
            end do
            if(res_atomname(ires,iatom).eq.'CA') then
c     $        res_atomname(ires,iatom).eq.'CB') then
              Cap_atomname(ires,step)='H'
              do k=1, 3
                 Cap(ires,step,k)=y(k)
              end do
            else
               Cap_atomname(ires,step)=
     $                 res_atomname(ires,iatom)(1:1)
            end if
         end if
      end do

      return
      end
c==========================================================
      subroutine ProCap_CH3(coord,res_atomname,res_atomnum,
     $                 ires,Cap,Cap_atomname,Cap_atomnum,
     $                 maxres,mxratm,step)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Cap(maxres,mxratm,3)
      character(len=4) Cap_atomname(maxres,mxratm)
      integer Cap_atomnum(maxres)
      integer Cap_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='CA'
      A2='N'
      AA='CC'
      call twopoints(coord,res_atomnum,res_atomname,
     $        ires-1,ires-1,A1,A2,maxres,mxratm,x,y)
      call bondlength(x,y,AA)
      A1='CA'
      A2='CB'
      call twopoints(coord,res_atomnum,res_atomname,
     $        ires-1,ires-1,A1,A2,maxres,mxratm,x,y1)
      call bondlength(x,y1,AA)

      do iatom=1, res_atomnum(ires-1)
         if(res_atomname(ires-1,iatom).eq.'C'.or.
     $        res_atomname(ires-1,iatom).eq.'O'.or.
     $        res_atomname(ires-1,iatom).eq.'CA'.or.
     $        res_atomname(ires-1,iatom).eq.'HA'.or.
     $        res_atomname(ires-1,iatom).eq.'HA1'.or.
     $        res_atomname(ires-1,iatom).eq.'CB'.or.
     $        res_atomname(ires-1,iatom).eq.'HA2'.or.
     $        res_atomname(ires-1,iatom).eq.'HA3'.or.
     $        res_atomname(ires-1,iatom).eq.'N') then
            step=step+1
            do k=1, 3
               Cap(ires,step,k)=coord(ires-1,iatom,k)
            end do
            if(res_atomname(ires-1,iatom).eq.'N') then
              Cap_atomname(ires,step)='H'
              do k=1, 3
                 Cap(ires,step,k)=y(k)
              end do
            end if
            if(res_atomname(ires-1,iatom).eq.'CB') then
              Cap_atomname(ires,step)='H'
              do k=1, 3
                 Cap(ires,step,k)=y1(k)
              end do
            end if
            if(res_atomname(ires-1,iatom).ne.'N'.and.
     $        res_atomname(ires-1,iatom).ne.'CB') then
              Cap_atomname(ires,step)=
     $                res_atomname(ires-1,iatom)(1:1)
            end if
         end if
      end do

      return
      end
C=================================================================
      subroutine ProCap_CH2R(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                maxres,mxratm,step)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Cap(maxres,mxratm,3)
      character(len=4) Cap_atomname(maxres,mxratm)
      integer Cap_atomnum(maxres)
      integer Cap_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='CA'
      A2='N'
      AA='CC'
      call twopoints(coord,res_atomnum,res_atomname,
     $        ires-1,ires-1,A1,A2,maxres,mxratm,x,y)
      call bondlength(x,y,AA)

      do iatom=1, res_atomnum(ires-1)
         if(res_atomname(ires-1,iatom).ne.'H'.and.
     $        res_atomname(ires-1,iatom).ne.'H1'.and.
     $        res_atomname(ires-1,iatom).ne.'H2'.and.
     $        res_atomname(ires-1,iatom).ne.'H3') then
            step=step+1
            do k=1, 3
               Cap(ires,step,k)=coord(ires-1,iatom,k)
            end do
            if(res_atomname(ires-1,iatom).eq.'N') then
              Cap_atomname(ires,step)='H'
              do k=1, 3
                 Cap(ires,step,k)=y(k)
              end do
            else
               Cap_atomname(ires,step)=
     $                res_atomname(ires-1,iatom)(1:1)
            end if
         end if
      end do

      return
      end
```

```
C=================================================================
      subroutine ProCap_CHRNH2(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                maxres,mxratm,step)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Cap(maxres,mxratm,3)
      character(len=4) Cap_atomname(maxres,mxratm)
      integer Cap_atomnum(maxres)
      integer Cap_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='N'
      A2='C'
      AA='CN'
      call twopoints(coord,res_atomnum,res_atomname,
     $        ires-1,ires-2,A1,A2,maxres,mxratm,x,y)
      call bondlength(x,y,AA)

      do iatom=1, res_atomnum(ires-1)
         step=step+1
         do k=1, 3
            Cap(ires,step,k)=coord(ires-1,iatom,k)
         end do
         Cap_atomname(ires,step)=
     $                res_atomname(ires-1,iatom)(1:1)
      end do

      do iatom=1, res_atomnum(ires-2)
         if(res_atomname(ires-2,iatom).eq.'C') then
            step=step+1
            do k=1, 3
               Cap(ires,step,k)=y(k)
            end do
            Cap_atomname(ires,step)='H'
         end if
      end do

      return
      end
C=================================================================
      subroutine ProCap_CHRNHCOH(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                maxres,mxratm,step)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Cap(maxres,mxratm,3)
      character(len=4) Cap_atomname(maxres,mxratm)
      integer Cap_atomnum(maxres)
      integer Cap_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='C'
      A2='CA'
      AA='CC'
      call twopoints(coord,res_atomnum,res_atomname,
     $        ires-2,ires-2,A1,A2,maxres,mxratm,x,y)
      call bondlength(x,y,AA)

      do iatom=1, res_atomnum(ires-1)
         step=step+1
         do k=1, 3
            Cap(ires,step,k)=coord(ires-1,iatom,k)
         end do
         Cap_atomname(ires,step)=
     $                res_atomname(ires-1,iatom)(1:1)
      end do

      do iatom=1, res_atomnum(ires-2)
         if(res_atomname(ires-2,iatom).eq.'C'.or.
     $        res_atomname(ires-2,iatom).eq.'O'.or.
     $        res_atomname(ires-2,iatom).eq.'CA') then
            step=step+1
            do k=1, 3
               Cap(ires,step,k)=coord(ires-2,iatom,k)
            end do
            if(res_atomname(ires-2,iatom).eq.'CA') then
              Cap_atomname(ires,step)='H'
              do k=1, 3
                 Cap(ires,step,k)=y(k)
              end do
            else
            Cap_atomname(ires,step)=
     $                 res_atomname(ires-2,iatom)(1:1)
            end if
         end if
      end do

      return
      end
c=================================================================
      subroutine readpdb(coord,res_atomname,res_atomnum,res_name,
     $                charge,endcharge,numres,maxres,mxratm)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)



      integer charge(maxres)
      integer endcharge(2)
      integer atomnum(maxres,mxratm)

      character (len=80) filepdb
      integer iarg
      integer iargc
      integer ierror
      integer ilen
      integer input_unit
      integer ios
      integer ipxfargc
      integer lenc
      integer numarg
      integer numres


c*----------------------Open the PDB file---------------------*c
      call get_unit(input_unit)
      open(unit=input_unit,file='protein.pdb',status='old',iostat=ios)
      if (ios.ne.0) then
          write(*,*) ' '
          write(*,*) 'PDB_PRB - Fatal error!'
          write(*,*) 'Could not open the PDB file.'
          stop
      end if


c*-------------Read the information from the file-------------*c
      call pdb_read(coord,res_atomname,res_atomnum,res_name,
     $              atomnum,input_unit,numres,maxres,mxratm)


c*---------------------Close the PDB file---------------------*c
      close(unit=input_unit)


c*-----------------find charge for each residue---------------*c
      call find_charge(res_atomname,res_atomnum,res_name,
     $                 charge,endcharge,numres,maxres,mxratm)
```

```
c*--change coordinates according to the pdb file from Chem3D--*c
c       call change_coord(coord,res_atomnum,atomnum,numres,
c     $                   maxres,mxratm)


c*-------------------------find CMS---------------------------*c
c       call find_cms(coord,res_atomname,res_atomnum,numres,
c     $                   maxres,mxratm)


c*------after CMS transformation, print out the pdb file------*c
c       call print_pdb(coord,res_atomname,res_atomnum,res_name,
c     $                   numres,maxres,mxratm)

        return
        end
c=============================================================
        subroutine get_unit(iunit)

        integer i
        integer ios
        integer iunit
        logical lopen

        iunit=0
        do i=1, 500
           if(i.ne.5.and.i.ne.6) then
              inquire(unit=i,opened=lopen,iostat=ios)
              if(ios.eq.0) then
                 if(.not.lopen) then
                    iunit=i
                    return
                 end if
              end if
           end if
        end do

        return
        end
c=============================================================
        subroutine pdb_init(coord,res_atomname,res_atomnum,
     $                       numres,maxres,mxratm)

        integer maxres
        integer mxratm

        real*8 coord(maxres,mxratm,3)
        character(len=4) res_atomname(maxres,mxratm)
        character(len=4) res_name(maxres)
        integer res_atomnum(maxres)



        integer charge(maxres)
        integer numres

        coord(1:maxres,1:mxratm,1:3)=0.0
        res_atomnum(1:maxres)=0
        res_atomname(1:maxres,1:mxratm)=' '
        charge(1:maxres)=0
        numres=0
```

```
      return
      end
c==================================================================
      subroutine pdb_read(coord,res_atomname,res_atomnum,
   $          res_name,atomnum,input_unit,numres,maxres,mxratm)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)



      integer atomnum(maxres,mxratm)

      integer ratm
      integer ibase
      integer input_unit
      integer ios
      integer numres
      integer prvnumres
      character prvchn
      integer prvresno
      character(len=3) prvresname
      character(len=128) string

      integer iatom
      integer resno
      real*8 x
      real*8 y
      real*8 z
      real*8 occ
      real*8 temp

      character(len=4) w1
      character(len=4) atomname
      character altloc
      character(len=3) resname
      character chains
      character icode
      character(len=4) segid
      character(len=2) element
      character(len=2) charge

      logical s_eqi


c*------------------Initialize PDB information----------------*c
      call pdb_init(coord,res_atomname,res_atomnum,
   $                 numres,maxres,mxratm)

      ibase=0
      prvchn=' '
      prvresno=0
      prvresname=' '

      do
         read(input_unit,'(a)',iostat=ios) string
         if(ios.ne.0) then
            exit
         end if
         if(s_eqi(string(1:6),'ENDMDL')) then
            exit
         else if(s_eqi(string(1:4),'ATOM')) then
            read(string,
     $           '(a6,i5,1x,a4,a1,a3,1x,a1,i4,a1,3x,
     $             3f8.3,2f6.2,6x,a4,a2,a2)',
     $             iostat=ios)
     $           w1,iatom,atomname,altloc,resname,chains,resno,
     $            icode,x,y,z,occ,temp,segid,element,charge
            if(ios.ne.0) then
               exit
            end if


c*---Remove a possible initial blank in ATOMNAME or RESNAME---*c
            if(atomname(1:1).eq.' ') then
               atomname=atomname(2:)
            end if

            if(resname(1:1).eq.' ') then
               resname = resname(2:)
            end if

            if(atomname(1:1).eq.'1'.or.atomname(1:1).eq.'2'.or.
     $         atomname(1:1).eq.'3') then
               atomname=atomname(2:)
            end if


c*-----If necessary, increment the number of residues read----*c
            if(resno.ne.prvresno.or.resname.ne.prvresname
     $         .or.chains.ne.prvchn) then
               prvresno=resno
               prvresname=resname
               numres=numres+1
               prvchn=chains
            end if


c*--For each atom, store the atomic coordinate, and the name--*c
c*----and number of the residue to which the atom belongs-----*c
            if(1.le.numres.and.numres.le.maxres) then
               if(numres.ne.prvnumres) ratm=0
               ratm=ratm+1
               res_name(numres)=resname
               res_atomnum(numres)=ratm
               coord(numres,ratm,1)=x
               coord(numres,ratm,2)=y
               coord(numres,ratm,3)=z
               res_atomname(numres,ratm)=atomname
               atomnum(numres,ratm)=iatom
               prvnumres=numres
            end if
         end if
      end do

      return
      end
c=================================================================
      function s_eqi(strng1,strng2)

      integer i
      integer len1
      integer len2
      integer lenc
      logical s_eqi
      character s1
      character s2
      character (len=*) strng1
      character (len=*) strng2

      len1=len(strng1)
      len2=len(strng2)
      lenc=min(len1,len2)

      s_eqi=.false.

      do i=1, lenc
         s1=strng1(i:i)
         s2=strng2(i:i)
         call c_cap(s1)
         call c_cap(s2)
         if(s1.ne.s2) then
            return
         end if
      end do

      do i=lenc+1, len1
         if(strng1(i:i).ne.' ') then
            return
         end if
      end do

      do i=lenc+1, len2
         if(strng2(i:i).ne.' ') then
            return
         end if
      end do

      s_eqi=.true.

      return
      end
c===============================================================
      subroutine c_cap(c)
      character c
      integer itemp

      itemp = ichar ( c )
      if(97.le.itemp.and.itemp.le.122) then
         c=char(itemp-32)
      end if

      return
      end
c===============================================================
      subroutine find_cms(coord,res_atomname,res_atomnum,
     $                    numres,maxres,mxratm)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)



      real*8 am(maxres,mxratm)
      integer numres
      real*8 totalm
      real*8 x1
      real*8 x2
      real*8 x3

      do ires=1, numres
         do iatom=1, res_atomnum(ires)
            if(res_atomname(ires,iatom)(1:1).eq.'H')
     $               am(ires,iatom)=1837.15d0
            if(res_atomname(ires,iatom)(1:1).eq.'C')
     $               am(ires,iatom)=21874.66d0
            if(res_atomname(ires,iatom)(1:1).eq.'N')
     $               am(ires,iatom)=25526.04d0
            if(res_atomname(ires,iatom)(1:1).eq.'O')
     $               am(ires,iatom)=29156.94d0
            if(res_atomname(ires,iatom)(1:1).eq.'S')
     $               am(ires,iatom)=58444.168d0
         end do
      end do

      totalm=0.d0
      x1=0.d0
      x2=0.d0
      x3=0.d0

      do ires=1, numres
         do iatom=1, res_atomnum(ires)
            totalm=totalm+am(ires,iatom)
            x1=x1+am(ires,iatom)*coord(ires,iatom,1)
            x2=x2+am(ires,iatom)*coord(ires,iatom,2)
            x3=x3+am(ires,iatom)*coord(ires,iatom,3)
         end do
      end do

      x1=x1/totalm
      x2=x2/totalm
      x3=x3/totalm

c     print*,'******************************'
c     print*,'      Center of Mass          '
c     print*,'x=',x1
c     print*,'y=',x2
c     print*,'z=',x3
c     print*,'******************************'

c     x1=coord(27,17,1)
c     x2=coord(27,17,2)
c     x3=coord(27,17,3)

      x1=30.d0
      x2=14.d0
      x3=-14.d0

      do ires=1, numres
         do iatom=1, res_atomnum(ires)
            coord(ires,iatom,1)=coord(ires,iatom,1)-x1
            coord(ires,iatom,2)=coord(ires,iatom,2)-x2
            coord(ires,iatom,3)=coord(ires,iatom,3)-x3
         end do
      end do

      return
      end
c=================================================================
      subroutine find_charge(res_atomname,res_atomnum,res_name,
   $                  charge,endcharge,numres,maxres,mxratm)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)



      integer charge(maxres)
      integer endcharge(2)
      integer numres
      integer ires
      integer iatom
      integer count_H
      integer count_O
      logical chg

c*------------initialize the each residue's charge------------*c
      charge(1:maxres)=0
      chg=.true.

      do ires=1, numres
         count_H=0
         count_O=0
         chg=.true.
c*-------------for ASP residue to check the charge------------*c
         if(res_name(ires).eq.'ASP') then
            do iatom=1, res_atomnum(ires)
               if(res_atomname(ires,iatom)(1:2).eq.'HD') chg=.false.
            end do
            if(chg) charge(ires)=-1
         end if
c*-------------for GLU residue to check the charge------------*c
         if(res_name(ires).eq.'GLU') then
            do iatom=1, res_atomnum(ires)
               if(res_atomname(ires,iatom)(1:2).eq.'HE') chg=.false.
            end do
            if(chg) charge(ires)=-1
         end if
c*-------------for LYS residue to check the charge------------*c
         if(res_name(ires).eq.'LYS') then
            do iatom=1, res_atomnum(ires)
               if(res_atomname(ires,iatom)(1:2).eq.'HZ') count_H=count_H+1
            end do
            if(count_H.eq.3) charge(ires)=1
         end if
c*-------------for ARG residue to check the charge------------*c
         if(res_name(ires).eq.'ARG') then
            do iatom=1, res_atomnum(ires)
               if(res_atomname(ires,iatom)(1:2).eq.'HH') count_H=count_H+1
            end do
            if(count_H.eq.4) charge(ires)=1
         end if
c*-----------for HIS/HID residue to check the charge----------*c
         if(res_name(ires).eq.'HIS'.or.res_name(ires).eq.'HID') then
            do iatom=1, res_atomnum(ires)
               if(res_atomname(ires,iatom)(1:3).eq.'HD'.or.
     $            res_atomname(ires,iatom)(1:3).eq.'HE') count_H=count_H+1
            end do
            if(count_H.eq.4) charge(ires)=1
         end if
c*----determine the charge for N-end residue in the chain-----*c
         count_H=0
         count_O=0
         if(ires.eq.1) then
            if(res_name(ires).ne.'PRO') then
               do iatom=1, res_atomnum(ires)
                  if(res_atomname(ires,iatom)(1:2).eq.'H1'.or.
     $               res_atomname(ires,iatom)(1:2).eq.'H2'.or.
     $               res_atomname(ires,iatom)(1:2).eq.'H3'.or.
     $               res_atomname(ires,iatom)(1:2).eq.'1H'.or.
     $               res_atomname(ires,iatom)(1:2).eq.'2H'.or.
     $               res_atomname(ires,iatom)(1:2).eq.'3H'.or.
     $               res_atomname(ires,iatom)(1:2).eq.'H') count_H=count_H+1
               end do
               if(count_H.eq.3) endcharge(1)=1
            else
               do iatom=1, res_atomnum(ires)
                  if(res_atomname(ires,iatom)(1:3).eq.'1H'.or.
     $               res_atomname(ires,iatom)(1:3).eq.'2H'.or.
     $               res_atomname(ires,iatom)(1:3).eq.'H1'.or.
     $               res_atomname(ires,iatom)(1:3).eq.'H2'.or.
     $               res_atomname(ires,iatom)(1:3).eq.'H') count_H=count_H+1
               end do
               if(count_H.eq.2) endcharge(1)=1
            end if
         end if
c*----determine the charge for C-end residue in the chain-----*c
         if(ires.eq.numres) then
            do iatom=1, res_atomnum(ires)
               if(res_atomname(ires,iatom)(1:3).eq.'OXT') then
                  endcharge(2)=-1
               end if
            end do
         end if

      end do

      return
      end
c==========================================================
      subroutine change_coord(coord,res_atomnum,atomnum,numres,
     $                           maxres,mxratm)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)
      integer atomnum(maxres,mxratm)
      integer iatom
      integer numres
      integer ires
      character(len=6) w1
      character(len=4) atomname
      integer num(1000)
      real x(1000)
      real y(1000)
      real z(1000)
      integer i

      open(3333,file='HIV1.pdb',status='old')
      do iatom=1, 985
         read(3333,"(a6,i5,a3,16x,f8.3,f8.3,f8.3)")
     $        w1,num(iatom),atomname,x(iatom),y(iatom),z(iatom)
      end do

      do ires=1, numres
         do iatom=1, res_atomnum(ires)
            do i=1, 985
               if(atomnum(ires,iatom).eq.num(i)) then
                  coord(ires,iatom,1)=x(i)
                  coord(ires,iatom,2)=y(i)
                  coord(ires,iatom,3)=z(i)
               end if
            end do
         end do
      end do

      return
      end
c==================================================================
      subroutine print_pdb(coord,res_atomname,res_atomnum,
     $                 res_name,numres,maxres,mxratm)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)



      integer ires
      integer numres
      integer iatom
      real occ
      real temp
      integer atomnum

      occ=1.00
      temp=0.00
      atomnum=0

      do ires=1, numres
         do iatom=1, res_atomnum(ires)
            atomnum=atomnum+1
            if(res_atomname(ires,iatom)(1:1).eq.'1'.or.
     $        res_atomname(ires,iatom)(1:1).eq.'2'.or.
     $        res_atomname(ires,iatom)(1:1).eq.'3') then
```

```
              write(5000,"(a4,2x,i5,1x,a4,1x,a3,2x,i4,4x,3f8.3,2f6.2)")
     $          'ATOM',atomnum,res_atomname(ires,iatom),res_name(ires),
     $          ires,coord(ires,iatom,1:3),occ,temp
            else
              write(5000,"(a4,2x,i5,2x,a4,a3,2x,i4,4x,3f8.3,2f6.2)")
     $          'ATOM',atomnum,res_atomname(ires,iatom),res_name(ires),
     $          ires,coord(ires,iatom,1:3),occ,temp
            end if
         end do
      end do

      write(5000,"(a3,3x,i5,2x,a4,a3,2x,i4)") 'TER',atomnum+1,'',
     $                res_name(numres),numres
      write(5000,"(a3)") 'END'

      return
      end
C==========================================================================
C==========================================================================
      subroutine calselect(Theory,BasisSet,level)

      integer level
      character(len=20) Theory
      character(len=20) BasisSet

      write(*,*) 'Protein Cut levels:'
      write(*,*) '     1. CH3NH2        '
      write(*,*) '     2. CH2RNH2       '
      write(*,*) '     3. CH2RNHCOH     '
      write(*,*) '     4. NH2COCHRNH2   '
      write(*,*) '     5. NH2COCHRNHCOH'
      write(*,*) 'Your Choice:          '
      read(*,*) level
      write(*,*) 'You have chosen #',level,'cut method'
      write(*,*) 'Select Theory:'
      read(*,*) Theory
      write(*,*) 'You have chosen ', Theory
      write(*,*) 'Select BasisSet:'
      read(*,*) BasisSet
      write(*,*) 'You have chosen ', BasisSet

      return
      end
C==========================================================================
      subroutine current_residue(coord,res_atomname,res_atomnum,
     $                Frag,Frag_atomname,Frag_atomnum,
     $                ires,maxres,mxratm)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Frag(maxres,mxratm,3)
      character(len=4) Frag_atomname(maxres,mxratm)
      integer Frag_atomnum(maxres)
      integer Frag_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      do iatom=1, res_atomnum(ires)
         do k=1, 3
            Frag(ires,iatom,k)=coord(ires,iatom,k)
         end do
         Frag_atomname(ires,iatom)=
     $                 res_atomname(ires,iatom)(1:1)
      end do
      Frag_atomnum(ires)=res_atomnum(ires)

      return
      end
C===========================================================
      subroutine CH3next(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,jres,
     $                maxres,mxratm,step)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Frag(maxres,mxratm,3)
      character(len=4) Frag_atomname(maxres,mxratm)
      integer Frag_atomnum(maxres)
      integer Frag_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='CA'
      A2='C'
      AA='CC'
      call twopoints(coord,res_atomnum,res_atomname,
     $          ires+1,ires+1,A1,A2,maxres,mxratm,x,y)
        call bondlength(x,y,AA)
        A1='CA'
        A2='CB'
        call twopoints(coord,res_atomnum,res_atomname,
     $          ires+1,ires+1,A1,A2,maxres,mxratm,x,y1)
        call bondlength(x,y1,AA)
        step=0
        do iatom=1, res_atomnum(ires+1)
           if(res_atomname(ires+1,iatom).eq.'N'.or.
     $          res_atomname(ires+1,iatom).eq.'H'.or.
     $          res_atomname(ires+1,iatom).eq.'C'.or.
     $          res_atomname(ires+1,iatom).eq.'CA'.or.
     $          res_atomname(ires+1,iatom).eq.'HA'.or.
     $          res_atomname(ires+1,iatom).eq.'CB'.or.
     $          res_atomname(ires+1,iatom).eq.'HA2'.or.
     $          res_atomname(ires+1,iatom).eq.'HA3') then
              step=step+1
              do k=1, 3
                 Frag(jres,Frag_atomnum(jres)+step,k)=
     $                coord(ires+1,iatom,k)
              end do
              if(res_atomname(ires+1,iatom).eq.'C') then
                Frag_atomname(jres,Frag_atomnum(jres)+step)='H'
                do k=1, 3
                   Frag(jres,Frag_atomnum(jres)+step,k)=y(k)
                end do
              end if
              if(res_atomname(ires+1,iatom).eq.'CB') then
                Frag_atomname(jres,Frag_atomnum(jres)+step)='H'
                do k=1, 3
                   Frag(jres,Frag_atomnum(jres)+step,k)=y1(k)
                end do
              end if
              if(res_atomname(ires+1,iatom).ne.'C'.and.
     $          res_atomname(ires+1,iatom).ne.'CB') then
                Frag_atomname(jres,Frag_atomnum(jres)+step)=
     $                          res_atomname(ires+1,iatom)(1:1)
              end if
           end if
        end do

        Frag_atomnum(jres)=Frag_atomnum(jres)+step

        return
        end
C=============================================================
        subroutine CH2Rnext(coord,res_atomname,res_atomnum,ires,
     $                Frag,Frag_atomname,Frag_atomnum,jres,
     $                maxres,mxratm,step)

        integer maxres
        integer mxratm

        real*8 coord(maxres,mxratm,3)
        character(len=4) res_atomname(maxres,mxratm)
        character(len=4) res_name(maxres)
        integer res_atomnum(maxres)


        real*8 Frag(maxres,mxratm,3)
        character(len=4) Frag_atomname(maxres,mxratm)
        integer Frag_atomnum(maxres)
```

```
      integer Frag_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='CA'
      A2='C'
      AA='CC'
      call twopoints(coord,res_atomnum,res_atomname,
     $        ires+1,ires+1,A1,A2,maxres,mxratm,x,y)
      call bondlength(x,y,AA)
      step=0
      do iatom=1, res_atomnum(ires+1)
         if(res_atomname(ires+1,iatom).ne.'O'.and.
     $        res_atomname(ires+1,iatom).ne.'OXT') then
            step=step+1
            do k=1, 3
               Frag(jres,Frag_atomnum(jres)+step,k)=
     $              coord(ires+1,iatom,k)
            end do
            if(res_atomname(ires+1,iatom).eq.'C') then
              Frag_atomname(jres,Frag_atomnum(jres)+step)='H'
              do k=1, 3
                 Frag(jres,Frag_atomnum(jres)+step,k)=y(k)
              end do
            else
               Frag_atomname(jres,Frag_atomnum(jres)+step)=
     $              res_atomname(ires+1,iatom)(1:1)
            end if
          end if
      end do

      Frag_atomnum(jres)=Frag_atomnum(jres)+step

      return
      end
C=========================================================
      subroutine CONHnext(coord,res_atomname,res_atomnum,ires,
     $                 Frag,Frag_atomname,Frag_atomnum,jres,
     $                 maxres,mxratm,step)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Frag(maxres,mxratm,3)
      character(len=4) Frag_atomname(maxres,mxratm)
      integer Frag_atomnum(maxres)
      integer Frag_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='N'
      A2='CA'
      AA='CN'
      call twopoints(coord,res_atomnum,res_atomname,
     $         ires+2,ires+2,A1,A2,maxres,mxratm,x,y)
      call bondlength(x,y,AA)
      step=0
      do iatom=1, res_atomnum(ires+1)
         step=step+1
         do k=1, 3
            Frag(jres,Frag_atomnum(jres)+step,k)=
     $                coord(ires+1,iatom,k)
         end do
         Frag_atomname(jres,Frag_atomnum(jres)+step)=
     $                res_atomname(ires+1,iatom)(1:1)
      end do

      do iatom=1, res_atomnum(ires+2)
         if(res_atomname(ires+2,iatom).eq.'N'.or.
     $        res_atomname(ires+2,iatom).eq.'H'.or.
     $        res_atomname(ires+2,iatom).eq.'CA') then
            step=step+1
            do k=1, 3
               Frag(jres,Frag_atomnum(jres)+step,k)=
     $                coord(ires+2,iatom,k)
            end do
            if(res_atomname(ires+2,iatom).eq.'CA') then
              Frag_atomname(jres,Frag_atomnum(jres)+step)='H'
              do k=1, 3
                 Frag(jres,Frag_atomnum(jres)+step,k)=y(k)
              end do
            else
               Frag_atomname(jres,Frag_atomnum(jres)+step)=
     $                res_atomname(ires+2,iatom)(1:1)
            end if
          end if
      end do

      Frag_atomnum(jres)=Frag_atomnum(jres)+step

      return
      end
C===========================================================
      subroutine NH2prev(coord,res_atomname,res_atomnum,ires,
     $                    Frag,Frag_atomname,Frag_atomnum,jres,
     $                    maxres,mxratm,step)

        integer maxres
        integer mxratm

        real*8 coord(maxres,mxratm,3)
        character(len=4) res_atomname(maxres,mxratm)
        character(len=4) res_name(maxres)
        integer res_atomnum(maxres)


        real*8 Frag(maxres,mxratm,3)
        character(len=4) Frag_atomname(maxres,mxratm)
        integer Frag_atomnum(maxres)
        integer Frag_id(maxres)

        integer i
        integer j
        integer iatom
        integer k
        integer step
        integer ires
        integer jres
        real*8 x(3)
        real*8 y(3)
        real*8 y1(3)
        character(len=4) A1
        character(len=4) A2
        character(len=4) AA


        A1='N'
        A2='C'
        AA='CN'
        call twopoints(coord,res_atomnum,res_atomname,
     $          ires,ires-1,A1,A2,maxres,mxratm,x,y)
        call bondlength(x,y,AA)
        step=0

        do iatom=1, res_atomnum(ires-1)
           if(res_atomname(ires-1,iatom).eq.'C') then
              step=step+1
              do k=1, 3
                 Frag(jres,Frag_atomnum(jres)+step,k)=y(k)
              end do
              Frag_atomname(jres,Frag_atomnum(jres)+step)='H'
           end if
        end do

        Frag_atomnum(jres)=Frag_atomnum(jres)+step

        return
        end
C=========================================================
        subroutine NHCOprev(coord,res_atomname,res_atomnum,ires,
     $                    Frag,Frag_atomname,Frag_atomnum,jres,
     $                    maxres,mxratm,step)

        integer maxres
        integer mxratm

        real*8 coord(maxres,mxratm,3)
```

```
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Frag(maxres,mxratm,3)
      character(len=4) Frag_atomname(maxres,mxratm)
      integer Frag_atomnum(maxres)
      integer Frag_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='C'
      A2='CA'
      AA='CC'
      call twopoints(coord,res_atomnum,res_atomname,
     $        ires-1,ires-1,A1,A2,maxres,mxratm,x,y)
      call bondlength(x,y,AA)
      step=0
      do iatom=1, res_atomnum(ires-1)
         if(res_atomname(ires-1,iatom).eq.'C'.or.
     $        res_atomname(ires-1,iatom).eq.'O'.or.
     $        res_atomname(ires-1,iatom).eq.'CA') then
            step=step+1
            do k=1, 3
               Frag(jres,Frag_atomnum(jres)+step,k)=
     $               coord(ires-1,iatom,k)
            end do
            if(res_atomname(ires-1,iatom).eq.'CA') then
              Frag_atomname(jres,Frag_atomnum(jres)+step)='H'
              do k=1, 3
                 Frag(jres,Frag_atomnum(jres)+step,k)=y(k)
              end do
            else
               Frag_atomname(jres,Frag_atomnum(jres)+step)=
     $               res_atomname(ires-1,iatom)(1:1)
            end if
         end if
      end do

      Frag_atomnum(jres)=Frag_atomnum(jres)+step

      return
      end
C=========================================================
      subroutine Cap_CH3(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                maxres,mxratm,step)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Cap(maxres,mxratm,3)
      character(len=4) Cap_atomname(maxres,mxratm)
      integer Cap_atomnum(maxres)
      integer Cap_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='CA'
      A2='C'
      AA='CC'
      call twopoints(coord,res_atomnum,res_atomname,
     $       ires,ires,A1,A2,maxres,mxratm,x,y)
      call bondlength(x,y,AA)
      A1='CA'
      A2='CB'
      call twopoints(coord,res_atomnum,res_atomname,
     $       ires,ires,A1,A2,maxres,mxratm,x,y1)
      call bondlength(x,y1,AA)

      do iatom=1, res_atomnum(ires)
         if(res_atomname(ires,iatom).eq.'N'.or.
     $       res_atomname(ires,iatom).eq.'H'.or.
     $       res_atomname(ires,iatom).eq.'C'.or.
     $       res_atomname(ires,iatom).eq.'CA'.or.
     $       res_atomname(ires,iatom).eq.'HA'.or.
     $       res_atomname(ires,iatom).eq.'CB'.or.
     $       res_atomname(ires,iatom).eq.'HA2'.or.
     $       res_atomname(ires,iatom).eq.'HA3') then
            step=step+1
            do k=1, 3
               Cap(ires,step,k)=coord(ires,iatom,k)
            end do
            if(res_atomname(ires,iatom).eq.'C') then
              Cap_atomname(ires,step)='H'
              do k=1, 3
                 Cap(ires,step,k)=y(k)
              end do
            end if
            if(res_atomname(ires,iatom).eq.'CB') then
              Cap_atomname(ires,step)='H'
              do k=1, 3
                 Cap(ires,step,k)=y1(k)
              end do
            end if
            if(res_atomname(ires,iatom).ne.'C'.and.
     $         res_atomname(ires,iatom).ne.'CB') then
               Cap_atomname(ires,step)=
     $                res_atomname(ires,iatom)(1:1)
            end if
         end if
      end do

      return
      end
c=================================================================
      subroutine Cap_CH2R(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                maxres,mxratm,step)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Cap(maxres,mxratm,3)
      character(len=4) Cap_atomname(maxres,mxratm)
      integer Cap_atomnum(maxres)
      integer Cap_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='CA'
      A2='C'
      AA='CC'
      call twopoints(coord,res_atomnum,res_atomname,
     $        ires,ires,A1,A2,maxres,mxratm,x,y)
      call bondlength(x,y,AA)

      do iatom=1, res_atomnum(ires)
         if(res_atomname(ires,iatom).ne.'O'.and.
     $        res_atomname(ires,iatom).ne.'OXT') then
            step=step+1
            do k=1, 3
               Cap(ires,step,k)=coord(ires,iatom,k)
            end do
            if(res_atomname(ires,iatom).eq.'C') then
              Cap_atomname(ires,step)='H'
              do k=1, 3
```

```
                Cap(ires,step,k)=y(k)
             end do
           else
              Cap_atomname(ires,step)=
     $                     res_atomname(ires,iatom)(1:1)
           end if
        end if
      end do

      return
      end
C=================================================================
      subroutine Cap_NH2(coord,res_atomname,res_atomnum,
     $                   ires,Cap,Cap_atomname,Cap_atomnum,
     $                   maxres,mxratm,step)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Cap(maxres,mxratm,3)
      character(len=4) Cap_atomname(maxres,mxratm)
      integer Cap_atomnum(maxres)
      integer Cap_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='N'
      A2='C'
      AA='CN'
      call twopoints(coord,res_atomnum,res_atomname,
     $         ires,ires-1,A1,A2,maxres,mxratm,x,y)
      call bondlength(x,y,AA)

      do iatom=1, res_atomnum(ires-1)
         if(res_atomname(ires-1,iatom).eq.'C') then
            step=step+1
            do k=1, 3
               Cap(ires,step,k)=y(k)
            end do
            Cap_atomname(ires,step)='H'
         end if
      end do

      return
      end
C=============================================================
      subroutine Cap_NHCO(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                maxres,mxratm,step)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Cap(maxres,mxratm,3)
      character(len=4) Cap_atomname(maxres,mxratm)
      integer Cap_atomnum(maxres)
      integer Cap_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='C'
      A2='CA'
      AA='CC'
      call twopoints(coord,res_atomnum,res_atomname,
     $        ires-1,ires-1,A1,A2,maxres,mxratm,x,y)
      call bondlength(x,y,AA)

      do iatom=1, res_atomnum(ires-1)
         if(res_atomname(ires-1,iatom).eq.'C'.or.
     $        res_atomname(ires-1,iatom).eq.'O'.or.
     $        res_atomname(ires-1,iatom).eq.'CA') then
            step=step+1
            do k=1, 3
               Cap(ires,step,k)=coord(ires-1,iatom,k)
            end do
            if(res_atomname(ires-1,iatom).eq.'CA') then
              Cap_atomname(ires,step)='H'
              do k=1, 3
                 Cap(ires,step,k)=y(k)
              end do
            else
               Cap_atomname(ires,step)=
     $                 res_atomname(ires-1,iatom)(1:1)
            end if
         end if
      end do

      return
      end
C===========================================================================
      subroutine Cap_CONH(coord,res_atomname,res_atomnum,
     $                ires,Cap,Cap_atomname,Cap_atomnum,
     $                maxres,mxratm,step)

      integer maxres
      integer mxratm

      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Cap(maxres,mxratm,3)
      character(len=4) Cap_atomname(maxres,mxratm)
      integer Cap_atomnum(maxres)
      integer Cap_id(maxres)

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      A1='N'
      A2='CA'
      AA='CN'
      call twopoints(coord,res_atomnum,res_atomname,
     $          ires+1,ires+1,A1,A2,maxres,mxratm,x,y)
      call bondlength(x,y,AA)

      do iatom=1, res_atomnum(ires)
         step=step+1
         do k=1, 3
            Cap(ires,step,k)=coord(ires,iatom,k)
         end do
         Cap_atomname(ires,step)=
     $                res_atomname(ires,iatom)(1:1)
      end do

      do iatom=1, res_atomnum(ires+1)
         if(res_atomname(ires+1,iatom).eq.'N'.or.
     $        res_atomname(ires+1,iatom).eq.'H'.or.
     $        res_atomname(ires+1,iatom).eq.'CA') then
            step=step+1
            do k=1, 3
               Cap(ires,step,k)=coord(ires+1,iatom,k)
            end do
            if(res_atomname(ires+1,iatom).eq.'CA') then
              Cap_atomname(ires,step)='H'
              do k=1, 3
                 Cap(ires,step,k)=y(k)
            end do
          else
            Cap_atomname(ires,step)=
     $                 res_atomname(ires+1,iatom)(1:1)
          end if
        end if
      end do

      return
      end
C=================================================================
      subroutine get_pccharge(charge,endcharge,pcharge,ccharge,
     $                        res_name,numres,maxres,level,cut)

      character(len=40) Theory
      integer level
      integer error
      integer cut
      integer numres
      integer ncpu

      integer maxres
      integer charge(maxres)
      integer endcharge(2)
      integer pcharge(maxres)
      integer ccharge(maxres)
      integer ires
      character(len=4) res_name(maxres)

      if(level.eq.1) then
         do ires=1, numres
            if(ires.eq.1) pcharge(ires)=charge(ires)
     $                        +endcharge(1)
            if(ires.eq.numres) pcharge(ires)=charge(ires)
     $                        +endcharge(2)
            if(1.lt.ires.and.ires.lt.numres) pcharge(ires)=
     $                        charge(ires)
         end do
      end if

      if(level.eq.2.or.level.eq.3.or.level.eq.4.or.
     $                        level.eq.5) then
         do ires=1, numres
csssssssssssssssss cut=0 i.e. cut CA-N bond ssssssssssssssssc
            if(cut.eq.0) then
               if(ires.eq.1) pcharge(ires)=charge(ires)+
     $                        charge(ires+1)+endcharge(1)
               if(ires.eq.numres) pcharge(ires)=charge(ires)+
     $                        endcharge(2)
               if(1.lt.ires.and.ires.lt.numres) pcharge(ires)=
     $                        charge(ires)+charge(ires+1)
            end if
csssssssssssssssss cut=1 i.e. cut CA-C bond ssssssssssssssssc
            if(cut.eq.1) then
               if(ires.eq.1) pcharge(ires)=charge(ires)
     $                        +endcharge(1)
               if(ires.eq.numres) pcharge(ires)=charge(ires)
     $                        +charge(ires-1)+endcharge(2)
               if(1.lt.ires.and.ires.lt.numres) pcharge(ires)=
     $                        charge(ires)+charge(ires-1)
            end if
csssssssssssssssss cut=2 i.e. cut C-N bond sssssssssssssssssc
            if(cut.eq.2) then
            if(level.eq.2) then
               if(ires.eq.1) pcharge(ires)=charge(ires)+
     $                     charge(ires+1)+endcharge(1)
               if(ires.eq.numres) pcharge(ires)=charge(ires)+
     $                     endcharge(2)
               if(1.lt.ires.and.ires.lt.numres) pcharge(ires)=
     $                     charge(ires)+charge(ires+1)
            end if
            if(level.eq.3) then
               if(ires.eq.1) pcharge(ires)=charge(ires)+
     $                     endcharge(1)
               if(ires.eq.numres) pcharge(ires)=charge(ires)+
     $                     charge(ires-1)+endcharge(2)
               if(1.lt.ires.and.ires.lt.numres) pcharge(ires)=
     $                     charge(ires)+charge(ires-1)
            end if
            if(level.eq.4) then
               if(ires.eq.1) pcharge(ires)=charge(ires)+
     $                     charge(ires+1)+endcharge(1)
               if(ires.eq.numres) pcharge(ires)=charge(ires)+
     $                     charge(ires-1)+endcharge(2)
               if(1.lt.ires.and.ires.lt.numres) pcharge(ires)=
     $                 charge(ires)+charge(ires+1)+charge(ires-1)
            end if
           end if
         end do
       end if
csssssssssssssssssssssssssssssssssssssssssssssssssssssssssssssss
       do ires=2, numres
         if(level.eq.1) then
            ccharge(ires)=0
         else
            if(cut.eq.0) ccharge(ires)=charge(ires)
            if(cut.eq.1) ccharge(ires)=charge(ires-1)
            if(cut.eq.2) then
               if(level.eq.2) ccharge(ires)=charge(ires)
               if(level.eq.3) ccharge(ires)=charge(ires-1)
               if(level.eq.4) ccharge(ires)=charge(ires)+
     $                     charge(ires-1)
            end if
         end if
       end do

       if(res_name(1).eq.'PRO') then
c         pcharge(2)=charge(2)+charge(1)+endcharge(1)
c         ccharge(2)=ccharge(2)+endcharge(1)
       end if

       if(res_name(numres).eq.'PRO') then
         pcharge(numres-1)=charge(numres-1)+charge(numres)
     $             +endcharge(numres)
         ccharge(numres)=ccharge(numres)+endcharge(numres)
       end if

       return
       end
c================================================================
       subroutine combineunits(coord,res_atomname,res_atomnum,
     $         Corr,Corr_atomname,Corr_atomnum,ires,level,
     $         charge,endcharge,Corr_charge,maxres,mxratm,numres)

       integer maxres
       integer mxratm
```

```
      real*8 coord(maxres,mxratm,3)
      character(len=4) res_atomname(maxres,mxratm)
      character(len=4) res_name(maxres)
      integer res_atomnum(maxres)


      real*8 Corr(maxres,mxratm,3)
      character(len=4) Corr_atomname(maxres,mxratm)
      integer Corr_atomnum(maxres)
      integer Corr_charge(maxres)
      character(len=40) Theory
      integer level
      integer error
      integer cut
      integer numres
      integer ncpu
      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      integer charge(maxres)
      integer endcharge(2)

      do iatom=1, res_atomnum(ires)
         do k=1, 3
            Corr(ires,iatom,k)=coord(ires,iatom,k)
         end do
         Corr_atomname(ires,iatom)=
     $                res_atomname(ires,iatom)(1:1)
      end do

      do iatom=1, res_atomnum(ires+1)
         do k=1, 3
            Corr(ires,res_atomnum(ires)+iatom,k)=
     $                coord(ires+1,iatom,k)
         end do
         Corr_atomname(ires,res_atomnum(ires)+iatom)=
     $                res_atomname(ires+1,iatom)(1:1)
      end do

      Corr_atomnum(ires)=res_atomnum(ires)+
     $                res_atomnum(ires+1)

      if(ires.eq.1) then
         if(level.eq.1) then
            call CH3next(coord,res_atomname,res_atomnum,
     $                ires+1,Corr,Corr_atomname,Corr_atomnum,
     $                ires,maxres,mxratm,step)
         end if
         if(level.eq.2.or.level.eq.3) then
            call CH2Rnext(coord,res_atomname,res_atomnum,
$                  ires+1,Corr,Corr_atomname,Corr_atomnum,
$                  ires,maxres,mxratm,step)
      end if
      if(level.eq.4.or.level.eq.5) then
         call CONHnext(coord,res_atomname,res_atomnum,
$                  ires+1,Corr,Corr_atomname,Corr_atomnum,
$                  ires,maxres,mxratm,step)
      end if
   end if

   if(1.lt.ires.and.ires.lt.(numres-1)) then
      if(level.eq.1) then
         call NH2prev(coord,res_atomname,res_atomnum,
$                  ires,Corr,Corr_atomname,Corr_atomnum,
$                  ires,maxres,mxratm,step)
         call CH3next(coord,res_atomname,res_atomnum,
$                  ires+1,Corr,Corr_atomname,Corr_atomnum,
$                  ires,maxres,mxratm,step)
      end if
      if(level.eq.2) then
         call NH2prev(coord,res_atomname,res_atomnum,
$                  ires,Corr,Corr_atomname,Corr_atomnum,
$                  ires,maxres,mxratm,step)
         call CH2Rnext(coord,res_atomname,res_atomnum,
$                  ires+1,Corr,Corr_atomname,Corr_atomnum,
$                  ires,maxres,mxratm,step)
      end if
      if(level.eq.3) then
         call NHCOprev(coord,res_atomname,res_atomnum,
$                  ires,Corr,Corr_atomname,Corr_atomnum,
$                  ires,maxres,mxratm,step)
         call CH2Rnext(coord,res_atomname,res_atomnum,
$                  ires+1,Corr,Corr_atomname,Corr_atomnum,
$                  ires,maxres,mxratm,step)
      end if
      if(level.eq.4) then
         call NH2prev(coord,res_atomname,res_atomnum,
$                  ires,Corr,Corr_atomname,Corr_atomnum,
$                  ires,maxres,mxratm,step)
         if(ires.eq.(numres-2)) then
            call CH2Rnext(coord,res_atomname,res_atomnum,
$                  ires+1,Corr,Corr_atomname,Corr_atomnum,
$                  ires,maxres,mxratm,step)
         else
            call CONHnext(coord,res_atomname,res_atomnum,
$                  ires+1,Corr,Corr_atomname,Corr_atomnum,
$                  ires,maxres,mxratm,step)
         end if
      end if
      if(level.eq.5) then
         call NHCOprev(coord,res_atomname,res_atomnum,
$                  ires,Corr,Corr_atomname,Corr_atomnum,
$                  ires,maxres,mxratm,step)
         if(ires.eq.(numres-2)) then
            call CH2Rnext(coord,res_atomname,res_atomnum,
$                  ires+1,Corr,Corr_atomname,Corr_atomnum,
$                  ires,maxres,mxratm,step)
         else
            call CONHnext(coord,res_atomname,res_atomnum,
$                  ires+1,Corr,Corr_atomname,Corr_atomnum,
$                  ires,maxres,mxratm,step)
         end if
      end if
      end if

      if(ires.eq.(numres-1)) then
         if(level.eq.1.or.level.eq.2.or.level.eq.4) then
            call NH2prev(coord,res_atomname,res_atomnum,
     $                   ires,Corr,Corr_atomname,Corr_atomnum,
     $                   ires,maxres,mxratm,step)
         end if
         if(level.eq.3.or.level.eq.5) then
            call NHCOprev(coord,res_atomname,res_atomnum,
     $                   ires,Corr,Corr_atomname,Corr_atomnum,
     $                   ires,maxres,mxratm,step)
         end if
      end if

      if(ires.eq.1) then
         if(level.eq.1) then
            Corr_charge(ires)=charge(ires)+charge(ires+1)
     $                          +endcharge(1)
         else
            Corr_charge(ires)=charge(ires)+charge(ires+1)
     $                          +charge(ires+2)+endcharge(1)
         end if
      end if

      if(ires.eq.(numres-1)) then
         Corr_charge(ires)=charge(ires)+charge(ires+1)
     $                          +endcharge(2)
      end if

      if(1.lt.ires.and.ires.lt.(numres-1)) then
         if(level.eq.1) then
            Corr_charge(ires)=charge(ires)+charge(ires+1)
         else
            Corr_charge(ires)=charge(ires)+charge(ires+1)
     $                          +charge(ires+2)
         end if
      end if

      return
      end
c=============================================================
      subroutine distance(x,y,dis)

      real*8 x(3)
      real*8 y(3)
      real*8 dis
      integer k

      do k=1, 3
         dis=dis+(x(k)-y(k))**2
      end do
      dis=dsqrt(dis)

      return
      end
c=============================================================
      subroutine mindis(tmp,tmp_atomnum,tmp_atomname,ires,numres,
     $                   ligand,ligand_atomnum,maxsize,maxres,
     $                   mxratm,min)

      integer maxres
      integer mxratm
      real*8 tmp(maxres,mxratm,3)
      character(len=4) tmp_atomname(maxres,mxratm)
      integer tmp_atomnum(maxres)
      integer tmp_id(maxres)
      integer maxsize

      character(len=4) atom_symbol(maxsize)
      real*8 ligand(maxsize,3)
      integer ligand_atomnum
      integer ligand_charge

      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      integer numres
      real*8 dis
      real*8 min

      min=10000.d0
      do iatom=1, tmp_atomnum(ires)
         do j=1, ligand_atomnum
            do k=1, 3
               x(k)=ligand(j,k)
            end do
            do k=1, 3
               y(k)=tmp(ires,iatom,k)
            end do
            call distance(x,y,dis)
            if(min.ge.dis) then
               min=dis
            end if
        end do
      end do

      return
      end
c=============================================================
      subroutine selectgroup(A_id,ires,nopolar,polar,charged,
     $                  min,maxres,id)

      integer maxres

      integer ires
      integer A_id(maxres)
      integer id
      real*8 min
      real nopolar
      real polar
      real charged
      id=1
      if(A_id(ires).eq.1) then
         if(min.gt.nopolar) id=0
      end if

      if(A_id(ires).eq.-1) then
         if(min.gt.polar) id=0
      end if

      if(A_id(ires).eq.0) then
         if(min.gt.charged) id=0
      end if

      return
      end
C=============================================================
      subroutine polarity(res_name,Frag_id,Cap_id,numres,
     $                maxres,level,cut)

      integer maxres
      character(len=40) Theory
      integer level
      integer error
      integer cut
      integer numres
      integer ncpu

      character(len=4) res_name(maxres)
      integer Frag_id(maxres)
      integer Cap_id(maxres)
      integer id(maxres)
      integer ires

      do ires=1, numres
         if(res_name(ires).eq.'ALA'.or.
     $        res_name(ires).eq.'VAL'.or.
     $        res_name(ires).eq.'LEU'.or.
     $        res_name(ires).eq.'ILE'.or.
     $        res_name(ires).eq.'PRO'.or.
     $        res_name(ires).eq.'MET'.or.
     $        res_name(ires).eq.'PHE'.or.
     $        res_name(ires).eq.'TRP') id(ires)=1

         if(res_name(ires).eq.'GLY'.or.
     $        res_name(ires).eq.'SER'.or.
     $        res_name(ires).eq.'THR'.or.
     $        res_name(ires).eq.'CYS'.or.
     $        res_name(ires).eq.'CYX'.or.
     $        res_name(ires).eq.'ASN'.or.
     $        res_name(ires).eq.'GLN'.or.
     $        res_name(ires).eq.'TYR') id(ires)=-1

         if(res_name(ires).eq.'ASP'.or.
     $        res_name(ires).eq.'GLU'.or.
     $        res_name(ires).eq.'LYS'.or.
     $        res_name(ires).eq.'ARG'.or.
     $        res_name(ires).eq.'HIS'.or.
     $        res_name(ires).eq.'HID') id(ires)=0

      end do

      id(1)=0
      id(numres)=0
```

```
csssssssssssssssss cut=0 i.e. cut CA-N bond ssssssssssssssssssc
        if(cut.eq.0) then
           if(level.eq.1) then
              do ires=1, numres
                 Frag_id(ires)=id(ires)
                 Cap_id(ires)=1
              end do
           else
              do ires=1, numres-1
                 if(id(ires).eq.-1.and.id(ires+1).eq.-1) then
                    Frag_id(ires)=-1
                 else
                    Frag_id(ires)=id(ires)*id(ires+1)
                 end if
                 Cap_id(ires+1)=id(ires+1)
              end do
              Frag_id(numres)=id(numres)
           end if
        end if
csssssssssssssssss cut=1 i.e. cut CA-C bond ssssssssssssssssssc
        if(cut.eq.1) then
           if(level.eq.1) then
              do ires=1, numres
                 Frag_id(ires)=id(ires)
                 Cap_id(ires)=1
              end do
           else
              do ires=2, numres
                 if(id(ires).eq.-1.and.id(ires-1).eq.-1) then
                    Frag_id(ires)=-1
                 else
                    Frag_id(ires)=id(ires)*id(ires-1)
                 end if
                 Cap_id(ires)=id(ires-1)
              end do
              Frag_id(1)=id(1)
           end if
        end if
csssssssssssssssss cut=2 i.e. cut C-N bond sssssssssssssssssssc
        if(cut.eq.2) then
           if(level.eq.1) then
              do ires=1, numres
                 Frag_id(ires)=id(ires)
                 Cap_id(ires)=1
              end do
           end if
           if(level.eq.2) then
              do ires=1, numres-1
                 if(id(ires).eq.-1.and.id(ires+1).eq.-1) then
                    Frag_id(ires)=-1
                 else
                    Frag_id(ires)=id(ires)*id(ires+1)
                 end if
                 Cap_id(ires+1)=id(ires+1)
              end do
              Frag_id(numres)=id(numres)
           end if
           if(level.eq.3) then
              do ires=2, numres
                 if(id(ires).eq.-1.and.id(ires-1).eq.-1) then
                    Frag_id(ires)=-1
                 else
                    Frag_id(ires)=id(ires)*id(ires-1)
```

```
                end if
                Cap_id(ires)=id(ires-1)
             end do
             Frag_id(1)=id(1)
          end if
          if(level.eq.4) then
c            do ires=2, numres-1
c               if(id(ires).eq.-1.and.id(ires-1).eq.-1) then
c                  Frag_id(ires)=-1
c               else
c                  Frag_id(ires)=id(ires-1)*id(ires)*id(ires+1)
c               end if
c               Cap_id(ires)=id(ires-1)*id(ires)
c             end do
c             Frag_id(1)=id(1)*id(2)
c            Frag_id(numres)=id(numres)*id(numres-1)
          end if

       end if

       return
       end
C=================================================================
       subroutine bondlength(x1,x2,id)
       implicit real*8(a-h,o-z)
       dimension x1(3),x2(3),y(2,3),z(2,3)
       character(len=4) id

       PI=dacos(-1.d0)
       bond=0.d0
       do i=1, 3
          bond=bond+(x1(i)-x2(i))**2
       end do
       bond=dsqrt(bond)

       do j=1, 3
          y(2,j)=x2(j)-x1(j)
          y(1,j)=0.d0
       end do

       theta=y(2,3)/bond
       if(theta.ge.1.d0) then
       theta=1.d0
       end if
       theta=dacos(theta)

       phi=abs(y(2,1))/bond/dsin(theta)
       if(phi.ge.1.d0) then
       phi=1.d0
       end if
       phi=dacos(phi)

       if(y(2,1).gt.0.d0.and.y(2,2).gt.0.d0) phi=phi
       if(y(2,1).lt.0.d0.and.y(2,2).gt.0.d0) phi=PI-phi
       if(y(2,1).lt.0.d0.and.y(2,2).lt.0.d0) phi=PI+phi
       if(y(2,1).gt.0.d0.and.y(2,2).lt.0.d0) phi=2.d0*PI-phi

       if(id.eq.'CC') bond=1.09d0
       if(id.eq.'CN') bond=1.01d0

       y(2,1)=bond*dsin(theta)*dcos(phi)
       y(2,2)=bond*dsin(theta)*dsin(phi)
       y(2,3)=bond*dcos(theta)
```

```
      do j=1, 3
         x2(j)=y(2,j)+x1(j)
      end do

      return
      end
c=============================================================
      subroutine twopoints(tmp,tmp_atomnum,tmp_atomname,ires,
     $                jres,atomi,atomj,maxres,mxratm,x,y)

      integer maxres
      integer mxratm

      real*8 tmp(maxres,mxratm,3)
      character(len=4) tmp_atomname(maxres,mxratm)
      integer tmp_atomnum(maxres)
      integer tmp_id(maxres)
      integer i
      integer j
      integer iatom
      integer k
      integer step
      integer ires
      integer jres
      real*8 x(3)
      real*8 y(3)
      real*8 y1(3)
      character(len=4) A1
      character(len=4) A2
      character(len=4) AA


      character(len=4) atomi
      character(len=4) atomj

      do iatom=1, tmp_atomnum(ires)
         if(tmp_atomname(ires,iatom).eq.atomi) then
            do k=1, 3
               x(k)=tmp(ires,iatom,k)
            end do
         end if
      end do

      do iatom=1, tmp_atomnum(jres)
         if(tmp_atomname(jres,iatom).eq.atomj) then
            do k=1, 3
               y(k)=tmp(jres,iatom,k)
            end do
         end if
      end do

      return
      end
c=============================================================
```